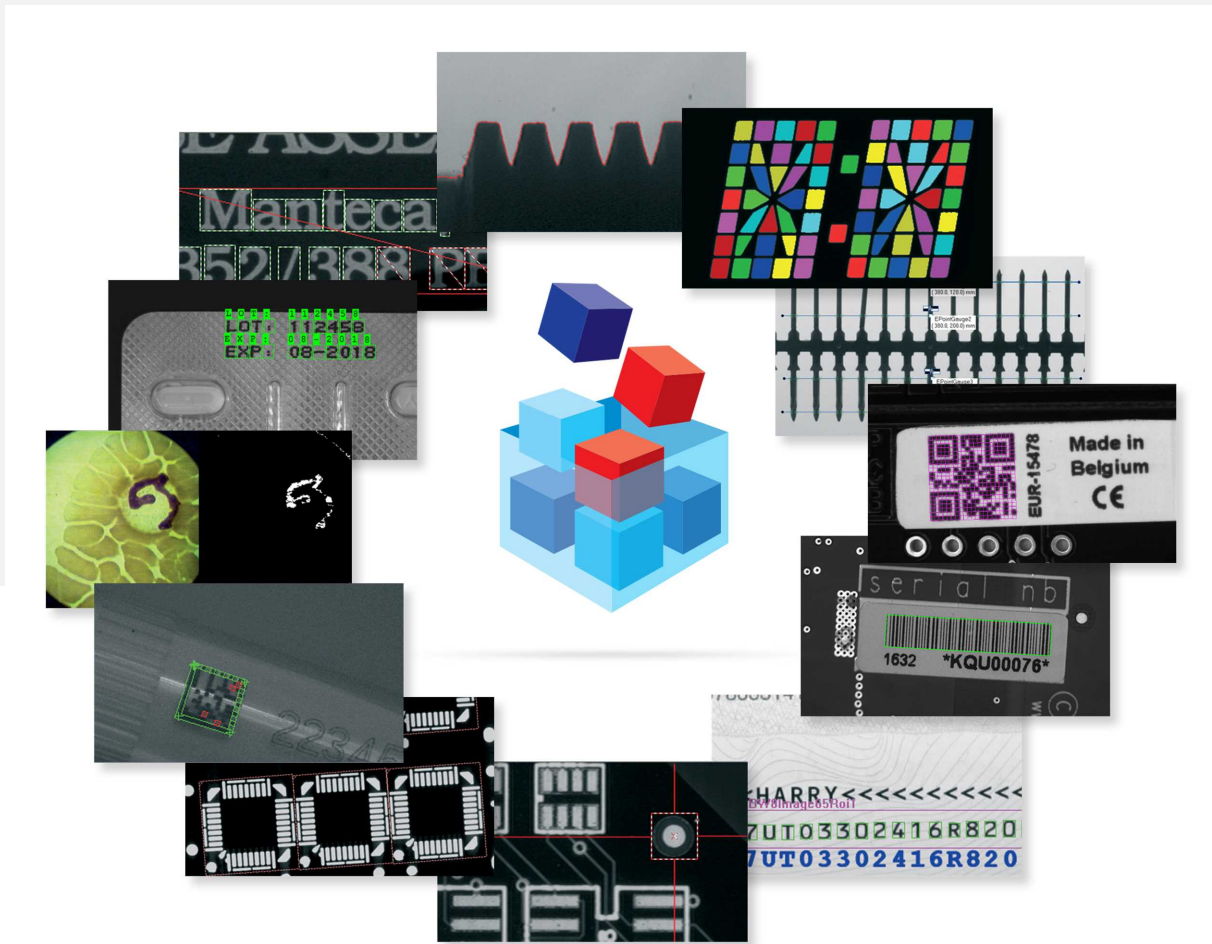


Open eVision



Terms of Use

EURESYS s.a. shall retain all property rights, title and interest of the documentation of the hardware and the software, and of the trademarks of EURESYS s.a.

All the names of companies and products mentioned in the documentation may be the trademarks of their respective owners.

The licensing, use, leasing, loaning, translation, reproduction, copying or modification of the hardware or the software, brands or documentation of EURESYS s.a. contained in this book, is not allowed without prior notice.

EURESYS s.a. may modify the product specification or change the information given in this documentation at any time, at its discretion, and without prior notice.

EURESYS s.a. shall not be liable for any loss of or damage to revenues, profits, goodwill, data, information systems or other special, incidental, indirect, consequential or punitive damages of any kind arising in connection with the use of the hardware or the software of EURESYS s.a. or resulting of omissions or errors in this documentation.

This documentation is provided with Open eVision 2.10.0 (doc build 1121).
© 2019 EURESYS s.a.

Contents

1. Pixel Accessors	76
1.1. EBW8PixelAccessor Class	76
EBW8PixelAccessor.EBW8PixelAccessor	76
EBW8PixelAccessor.GetPixel	77
EBW8PixelAccessor.SetPixel	77
1.2. EBW16PixelAccessor Class	78
EBW16PixelAccessor.EBW16PixelAccessor	78
EBW16PixelAccessor.GetPixel	79
EBW16PixelAccessor.SetPixel	79
1.3. EBW32PixelAccessor Class	80
EBW32PixelAccessor.EBW32PixelAccessor	80
EBW32PixelAccessor.GetPixel	81
EBW32PixelAccessor.SetPixel	81
1.4. EC15PixelAccessor Class	82
EC15PixelAccessor.EC15PixelAccessor	83
EC15PixelAccessor.GetPixel	83
EC15PixelAccessor.SetPixel	84
1.5. EC16PixelAccessor Class	84
EC16PixelAccessor.EC16PixelAccessor	85
EC16PixelAccessor.GetPixel	85
EC16PixelAccessor.SetPixel	86
1.6. EC24APixelAccessor Class	86
EC24APixelAccessor.EC24APixelAccessor	87
EC24APixelAccessor.GetPixel	87
EC24APixelAccessor.SetPixel	88
1.7. EC24PixelAccessor Class	88
EC24PixelAccessor.EC24PixelAccessor	89
EC24PixelAccessor.GetPixel	89
EC24PixelAccessor.SetPixel	90
2. Common	91
2.1. Easy Class	91
2.2. Image and ROI Classes	91
2.3. Region Classes	92
3. Libraries	93
3.1. Easy3D Library	93
3.2. Easy3DLaserLine Library	94
3.3. Easy3DObject Library	95
3.4. EasyImage Library	95
3.5. EasyColor Library	96
3.6. EasyObject Library	97
3.7. EasyMatch Library	98
3.8. EasyFind Library	98
3.9. EasyGauge Library	99
3.10. EasyOCR Library	99

3.11. EasyOCR2 Library	100
3.12. EasyOCV Library	101
3.13. EasyBarCode Library	101
3.14. EasyMatrixCode Library	102
3.15. EasyMatrixCode2 Library	102
3.16. EasyQRCode Library	103
3.17. EasyDeepLearning Library	103
3.18. Legacy	104
EasyObject Library (Legacy)	104
4. Classes	105
4.1. E3DAxisSystem Class	105
E3DAxisSystem.AxisX	107
E3DAxisSystem.AxisY	107
E3DAxisSystem.AxisZ	107
E3DAxisSystem.CheckIsNormal	108
E3DAxisSystem.CheckIsOrthogonal	108
E3DAxisSystem.CheckIsRightHanded	109
E3DAxisSystem.E3DAxisSystem	110
E3DAxisSystem.IsNormal	111
E3DAxisSystem.IsOrthogonal	111
E3DAxisSystem.IsRightHanded	111
E3DAxisSystem.NormX	112
E3DAxisSystem.NormY	112
E3DAxisSystem.NormZ	113
E3DAxisSystem.operator!=	113
E3DAxisSystem.operator=	113
E3DAxisSystem.operator==	114
E3DAxisSystem.Origin	114
E3DAxisSystem.Serialize	115
4.2. E3DBox Class	115
E3DBox.Axes	117
E3DBox.Center	117
E3DBox.E3DBox	118
E3DBox.Load	119
E3DBox.operator!=	120
E3DBox.operator=	120
E3DBox.operator==	121
E3DBox.Save	121
E3DBox.XAxis	122
E3DBox.XSize	122
E3DBox.XYQuadrangle	123
E3DBox.YAxis	123
E3DBox.YSize	123
E3DBox.ZAxis	124
E3DBox.ZSize	124
4.3. E3DObject Class	125
E3DObject.Area	128
E3DObject.AveragePosition	128
E3DObject.BasePlane	128
E3DObject.BaseTilt	129
E3DObject.BoundingBox	129
E3DObject.Draw	130
E3DObject.E3DObject	131

E3DObject.Length	131
E3DObject.Load	132
E3DObject.LocalHeight	132
E3DObject.LocalTilt	133
E3DObject.LocalTopPosition	133
E3DObject.NumPixels	133
E3DObject.operator=	134
E3DObject.operator==	134
E3DObject.Orientation	135
E3DObject.Plane	135
E3DObject.RectangleRegion	136
E3DObject.ReferenceHeight	136
E3DObject.ReferenceTilt	136
E3DObject.ReferenceTopPosition	137
E3DObject.Region	137
E3DObject.Save	138
E3DObject.Volume	138
E3DObject.Width	139
4.4. E3DObjectExtractor Class	139
E3DObjectExtractor.AddToMesh	142
E3DObjectExtractor.AreaRange	143
E3DObjectExtractor.AspectRatioRange	143
E3DObjectExtractor.Draw	143
E3DObjectExtractor.E3DObjectExtractor	144
E3DObjectExtractor.Extract	145
E3DObjectExtractor.LengthRange	146
E3DObjectExtractor.Load	147
E3DObjectExtractor.LocalHeightRange	147
E3DObjectExtractor.LocalTiltRange	148
E3DObjectExtractor.Objects	148
E3DObjectExtractor.operator=	149
E3DObjectExtractor.operator==	149
E3DObjectExtractor.OrientationRange	150
E3DObjectExtractor.ReferenceHeightRange	150
E3DObjectExtractor.ReferenceTiltRange	150
E3DObjectExtractor.Save	151
E3DObjectExtractor.VolumeRange	151
E3DObjectExtractor.WidthRange	152
4.5. E3DOrthonormalAxisSystem Class	152
E3DOrthonormalAxisSystem.E3DOrthonormalAxisSystem	153
E3DOrthonormalAxisSystem.operator=	154
4.6. E3DPlane Class	154
E3DPlane.Define	156
E3DPlane.DistanceTo	157
E3DPlane.E3DPlane	158
E3DPlane.Load	159
E3DPlane.Normal	159
E3DPlane.operator-	160
E3DPlane.operator+	160
E3DPlane.operator=	161
E3DPlane.ProjectPoint	161
E3DPlane.Save	162
E3DPlane.SignedDistanceFromOrigin	162
4.7. E3DRightOrthonormalAxisSystem Class	163
E3DRightOrthonormalAxisSystem.E3DRightOrthonormalAxisSystem	163

E3DRightOrthonormalAxisSystem.operator=	164
4.8. E3DTransformMatrix Class	165
E3DTransformMatrix.CreateAnisotropicScalingMatrix	167
E3DTransformMatrix.CreateIdentityMatrix	168
E3DTransformMatrix.CreateIsotropicScalingMatrix	168
E3DTransformMatrix.CreateOrthoBasis	169
E3DTransformMatrix.CreateOrthographicProjectionMatrix	170
E3DTransformMatrix.CreatePerspectiveProjectionMatrix	170
E3DTransformMatrix.CreateRotationXMatrix	171
E3DTransformMatrix.CreateRotationYMatrix	172
E3DTransformMatrix.CreateRotationZMatrix	172
E3DTransformMatrix.CreateTranslationMatrix	173
E3DTransformMatrix.E3DTransformMatrix	173
E3DTransformMatrix.GetOrthoBasis	175
E3DTransformMatrix.GetValue	176
E3DTransformMatrix.Inverse	177
E3DTransformMatrix.IsRigid	177
E3DTransformMatrix.Load	177
E3DTransformMatrix.operator!=	178
E3DTransformMatrix.operator*	178
E3DTransformMatrix.operator+	179
E3DTransformMatrix.operator=	180
E3DTransformMatrix.operator==	180
E3DTransformMatrix.Save	181
E3DTransformMatrix.SetValue	181
E3DTransformMatrix.Transpose	182
4.9. E3DViewer Class	182
E3DViewer.Colors	185
E3DViewer.ConfigureRenderSource	185
E3DViewer.E3DViewer	186
E3DViewer.GenerateColors	188
E3DViewer.HideFeatureFor3DObject	188
E3DViewer.HideFeatureForAll3DObjects	189
E3DViewer.PointSize	189
E3DViewer.Register3DObjects	190
E3DViewer.RemoveCurrent3DObjects	190
E3DViewer.RenderDecimationLevel	191
E3DViewer.RenderGrid	191
E3DViewer.RenderGridStep	192
E3DViewer.SetAutoRotate	192
E3DViewer.SetFeatureStyleFor3DObject	193
E3DViewer.SetFeatureStyleForAll3DObjects	193
E3DViewer.SetFocus	194
E3DViewer.SetPosition	194
E3DViewer.SetViewAngle	195
E3DViewer.SetViewTarget	196
E3DViewer.Show	196
E3DViewer.ShowFeatureFor3DObject	197
E3DViewer.ShowFeatureForAll3DObjects	197
E3DViewer.StopAutoRotate	198
E3DViewer.ViewDistance	198
E3DViewer.WireframeMode	199
4.10. EAffineTransformer Class	199
EAffineTransformer.AddAnisotropicScalingTransform	202
EAffineTransformer.AddIsotropicScalingTransform	203

EAffineTransformer.AddOrthographicProjectionTransform	203
EAffineTransformer.AddPerspectiveProjectionTransform	204
EAffineTransformer.AddRotationXTransform	204
EAffineTransformer.AddRotationYTransform	205
EAffineTransformer.AddRotationZTransform	205
EAffineTransformer.AddTransform	206
EAffineTransformer.AddTranslationTransform	206
EAffineTransformer.ApplyMatrix	207
EAffineTransformer.ApplyTransform	208
EAffineTransformer.CreateAnisotropicScalingMatrix	209
EAffineTransformer.CreateIdentityMatrix	210
EAffineTransformer.CreateIsotropicScalingMatrix	210
EAffineTransformer.CreateOrthographicProjectionMatrix	211
EAffineTransformer.CreatePerspectiveProjectionMatrix	211
EAffineTransformer.CreateRotationXMatrix	212
EAffineTransformer.CreateRotationYMatrix	212
EAffineTransformer.CreateRotationZMatrix	213
EAffineTransformer.CreateTranslationMatrix	213
EAffineTransformer.EAffineTransformer	214
EAffineTransformer.operator=	215
EAffineTransformer.Reset	215
EAffineTransformer.Transform	215
4.11. EAngleRectifier Class	216
EAngleRectifier.Rectify	216
4.12. Easy Class	217
Easy.AngleUnit	220
Easy.CheckLicense	220
Easy.CheckLicenses	221
Easy.CheckOemKey	221
Easy.CloseImageGraphicContext	222
Easy.DongleCount	223
Easy.FromRadians	223
Easy.GetBestMatchingImageType	224
Easy.GetDongleInternalSerialNumber	224
Easy.GetErrorText	225
Easy.MaxNumberOfProcessingThreads	225
Easy.NumberOfAvailableProcessorCores	226
Easy.OpenImageGraphicContext	226
Easy.Render3D	227
Easy.RenderColorHistogram	228
Easy.Resize	230
Easy.SetOemKey	231
Easy.StartTiming	232
Easy.StopTiming	232
Easy.Terminate	232
Easy.ToRadians	233
Easy.TrueTimingResolution	233
Easy.Version	234
4.13. EasyColor Class	234
EasyColor.AlphaBlend	241
EasyColor.AssignNearestClass	242
EasyColor.AssignNearestClassCenter	243
EasyColor.BayerToC24	244
EasyColor.C24ToBayer	245
EasyColor.CieAB	246

EasyColor.CieAG	246
EasyColor.CieAR	246
EasyColor.CieD50B	247
EasyColor.CieD50G	247
EasyColor.CieD50R	248
EasyColor.CieD55B	248
EasyColor.CieD55G	248
EasyColor.CieD55R	249
EasyColor.CieD65B	249
EasyColor.CieD65G	250
EasyColor.CieD65R	250
EasyColor.CieFB	250
EasyColor.CieFG	251
EasyColor.CieFR	251
EasyColor.ClassAverages	252
EasyColor.ClassVariances	253
EasyColor.CompensateNtscGamma	254
EasyColor.CompensatePalGamma	254
EasyColor.CompensateSmpteGamma	254
EasyColor.Compose	255
EasyColor.Decompose	256
EasyColor.Dequantize	257
EasyColor.DstQuantization	259
EasyColor.Format422To444	259
EasyColor.Format444To422	260
EasyColor.GetComponent	261
EasyColor.ImproveClassCenters	261
EasyColor.IshToRgb	262
EasyColor.LabToRgb	263
EasyColor.LabToXyz	264
EasyColor.LchToRgb	264
EasyColor.LshToRgb	265
EasyColor.LuvToRgb	266
EasyColor.LuvToXyz	267
EasyColor.NtscGamma	267
EasyColor.PalGamma	268
EasyColor.PseudoColor	268
EasyColor.Quantize	269
EasyColor.RegisterPlanes	271
EasyColor.RgbStandard	272
EasyColor.RgbToIsh	273
EasyColor.RgbToLab	273
EasyColor.RgbToLch	274
EasyColor.RgbToLsh	275
EasyColor.RgbToLuv	276
EasyColor.RgbToReducedXyz	276
EasyColor.RgbToVsh	277
EasyColor.RgbToXyz	278
EasyColor.RgbToYiq	279
EasyColor.RgbToYsh	279
EasyColor.RgbToYuv	280
EasyColor.SetComponent	281
EasyColor.SmpteGamma	282
EasyColor.SrcQuantization	282
EasyColor.Transform	283

EasyColor.TransformBayer	284
EasyColor.VshToRgb	285
EasyColor.XyzToLab	285
EasyColor.XyzToLuv	286
EasyColor.XyzToRgb	287
EasyColor.YiqToRgb	288
EasyColor.YshToRgb	288
EasyColor.YuvToRgb	289
4.14. EasyImage Class	290
EasyImage.AdaptiveThreshold	307
EasyImage.AlphaBlend	307
EasyImage.AnalyseHistogram	308
EasyImage.AnalyseHistogramBW16	309
EasyImage.Area	310
EasyImage.AreaDoubleThreshold	311
EasyImage.ArgumentImage	313
EasyImage.AutoThreshold	314
EasyImage.BiLevelBlackTopHatBox	315
EasyImage.BiLevelBlackTopHatDisk	316
EasyImage.BiLevelCloseBox	317
EasyImage.BiLevelCloseDisk	318
EasyImage.BiLevelDilateBox	318
EasyImage.BiLevelDilateDisk	319
EasyImage.BiLevelErodeBox	320
EasyImage.BiLevelErodeDisk	321
EasyImage.BiLevelMedian	321
EasyImage.BiLevelMorphoGradientBox	322
EasyImage.BiLevelMorphoGradientDisk	323
EasyImage.BiLevelOpenBox	324
EasyImage.BiLevelOpenDisk	325
EasyImage.BiLevelThick	326
EasyImage.BiLevelThin	327
EasyImage.BiLevelWhiteTopHatBox	328
EasyImage.BiLevelWhiteTopHatDisk	328
EasyImage.BinaryMoments	329
EasyImage.BlackTopHatBox	333
EasyImage.BlackTopHatDisk	334
EasyImage.CloseBox	335
EasyImage.CloseDisk	337
EasyImage.Contour	338
EasyImage.Convert	340
EasyImage.ConvertTo422	345
EasyImage.ConvolveGaussian	346
EasyImage.ConvolveGradient	348
EasyImage.ConvolveGradientX	348
EasyImage.ConvolveGradientY	349
EasyImage.ConvolveHighpass1	350
EasyImage.ConvolveHighpass2	351
EasyImage.ConvolveKernel	352
EasyImage.ConvolveLaplacian4	353
EasyImage.ConvolveLaplacian8	354
EasyImage.ConvolveLaplacianX	355
EasyImage.ConvolveLaplacianY	356
EasyImage.ConvolveLowpass1	357
EasyImage.ConvolveLowpass2	358

EasyImage.ConvolveLowpass3	359
EasyImage.ConvolvePrewitt	360
EasyImage.ConvolvePrewittX	361
EasyImage.ConvolvePrewittY	362
EasyImage.ConvolveRoberts	362
EasyImage.ConvolveSobel	363
EasyImage.ConvolveSobelX	364
EasyImage.ConvolveSobelY	365
EasyImage.ConvolveSymmetricKernel	366
EasyImage.ConvolveUniform	367
EasyImage.Copy	369
EasyImage.CumulateHistogram	371
EasyImage.DilateBox	372
EasyImage.DilateDisk	373
EasyImage.Distance	374
EasyImage.DoubleThreshold	375
EasyImage.Equalize	377
EasyImage.ErodeBox	378
EasyImage.ErodeDisk	379
EasyImage.Focusing	380
EasyImage.Gain	381
EasyImage.GainOffset	381
EasyImage.GetFrame	383
EasyImage.GetProfilePeaks	384
EasyImage.GradientScalar	385
EasyImage.GravityCenter	386
EasyImage.HDRFusion	388
EasyImage.Histogram	389
EasyImage.HistogramThreshold	391
EasyImage.HistogramThresholdBW16	392
EasyImage.HitAndMiss	393
EasyImage.HorizontalMirror	394
EasyImage.ImageToLineSegment	395
EasyImage.ImageToPath	397
EasyImage.IsodataThreshold	399
EasyImage.IsodataThresholdBW16	400
EasyImage.LinearTransform	400
EasyImage.LineSegmentToImage	402
EasyImage.LocalAverage	404
EasyImage.LocalDeviation	405
EasyImage.Lut	406
EasyImage.MatchFrames	408
EasyImage.Median	409
EasyImage.ModulusImage	410
EasyImage.MorphoGradientBox	411
EasyImage.MorphoGradientDisk	412
EasyImage.Normalize	413
EasyImage.Offset	414
EasyImage.OpenBox	415
EasyImage.OpenDisk	417
EasyImage.Oper	418
EasyImage.Overlay	422
EasyImage.OverlayColor	424
EasyImage.PathToImage	425
EasyImage.PixelAverage	426

EasyImage.PixelCompare	428
EasyImage.PixelCount	429
EasyImage.PixelMax	433
EasyImage.PixelMaxBW16	434
EasyImage.PixelMaxBW8	435
EasyImage.PixelMin	436
EasyImage.PixelMinBW16	437
EasyImage.PixelMinBW8	437
EasyImage.PixelStat	438
EasyImage.PixelStatBW16	440
EasyImage.PixelStatBW8	440
EasyImage.PixelStdDev	441
EasyImage.PixelVariance	445
EasyImage.ProfileDerivative	448
EasyImage.ProjectOnAColumn	449
EasyImage.ProjectOnARow	451
EasyImage.RealignFrame	452
EasyImage.RebuildFrame	454
EasyImage.RecursiveAverage	455
EasyImage.Register	456
EasyImage.RmsNoise	461
EasyImage.ScaleRotate	463
EasyImage.SetCircleWarp	465
EasyImage.SetFrame	466
EasyImage.SetRecursiveAverageLUT	467
EasyImage.SetupEqualize	468
EasyImage.Shrink	469
EasyImage.SignalNoiseRatio	470
EasyImage.SwapFrames	472
EasyImage.Thick	472
EasyImage.Thin	474
EasyImage.ThreeLevelsMinResidueThreshold	475
EasyImage.Threshold	476
EasyImage.TwoLevelsMinResidueThreshold	481
EasyImage.Uniformize	482
EasyImage.VerticalMirror	486
EasyImage.Warp	487
EasyImage.WeightedMoments	488
EasyImage.WhiteTopHatBox	498
EasyImage.WhiteTopHatDisk	499
4.15. EasyObject Class	500
EasyObject.ContourArea	501
EasyObject.ContourGravityCenter	501
EasyObject.ContourInertia	502
EasyObject.IsFloatFeature	503
EasyObject.IsIntegerFeature	504
EasyObject.IsUnsignedIntegerFeature	504
4.16. EBarCode Class	505
EBarCode.AdditionalSymbologies	508
EBarCode.Decode	509
EBarCode.Detect	509
EBarCode.Drag	510
EBarCode.Draw	511
EBarCode.DrawWithCurrentPen	512
EBarCode.EBarCode	512

EBarCode.GetDecodedAngle	513
EBarCode.GetDecodedDirection	514
EBarCode.GetDecodedRectangle	515
EBarCode.GetDecodedSymbology	515
EBarCode.GetSymbologyName	516
EBarCode.HitTest	516
EBarCode.KnownLocation	517
EBarCode.KnownModule	517
EBarCode.Module	518
EBarCode.NumDecodedSymbologies	518
EBarCode.NumEnabledSymbologies	519
EBarCode.Read	519
EBarCode.Rectangle	520
EBarCode.RelativeReadingSizeX	520
EBarCode.RelativeReadingSizeY	521
EBarCode.RelativeReadingX	521
EBarCode.RelativeReadingY	521
EBarCode.SetReadingCenter	522
EBarCode.SetReadingSize	522
EBarCode.StandardSymbologies	523
EBarCode.ThicknessRatio	523
EBarCode.VerifyChecksum	524
4.17. EBaseROI Class	525
EBaseROI.Attach	530
EBaseROI.Author	531
EBaseROI.BaseTopParent	531
EBaseROI.BitsPerPixel	531
EBaseROI.ColorSystem	532
EBaseROI.ColPitch	532
EBaseROI.Comment	533
EBaseROI.CopyTo	533
EBaseROI.CropToImage	534
EBaseROI.Date	534
EBaseROI.Drag	535
EBaseROI.Draw	536
EBaseROI.DrawFrame	538
EBaseROI.DrawFrameWithCurrentPen	540
EBaseROI.GetImagePtr	541
EBaseROI.GetSubBaseROIs	542
EBaseROI.HasSubROI	543
EBaseROI.Height	543
EBaseROI.HitTest	544
EBaseROI.IsAnROI	545
EBaseROI.IsVoid	545
EBaseROI.Load	545
EBaseROI.OrgX	546
EBaseROI.OrgY	547
EBaseROI.Parent	547
EBaseROI.PlanesPerPixel	548
EBaseROI.RowPitch	548
EBaseROI.Save	548
EBaseROI.SaveJpeg	549
EBaseROI.SaveJpeg2K	550
EBaseROI.SavePng	550
EBaseROI.Serialize	551

EBaseROI.SetImagePtr	552
EBaseROI.SetPlacement	553
EBaseROI.SetSize	553
EBaseROI.Title	554
EBaseROI.TotalHeight	555
EBaseROI.TotalOrgX	555
EBaseROI.TotalOrgY	556
EBaseROI.TotalWidth	556
EBaseROI.Type	557
EBaseROI.Width	557
4.18. EBinaryImageSegmenter Class	557
4.19. EBW16PathVector Class	558
EBW16PathVector.AddElement	559
EBW16PathVector.Closed	560
EBW16PathVector.Draw	560
EBW16PathVector.DrawWithCurrentPen	562
EBW16PathVector.EBW16PathVector	563
EBW16PathVector.GetElement	563
EBW16PathVector.operator[]	564
EBW16PathVector.operator=	564
EBW16PathVector.RawDataPtr	565
EBW16PathVector.SetElement	565
4.20. EBW16PixelAccessor Class	566
EBW16PixelAccessor.EBW16PixelAccessor	567
EBW16PixelAccessor.GetPixel	567
EBW16PixelAccessor.SetPixel	568
4.21. EBW16Vector Class	568
EBW16Vector.AddElement	570
EBW16Vector.Draw	570
EBW16Vector.DrawWithCurrentPen	572
EBW16Vector.EBW16Vector	573
EBW16Vector.GetElement	573
EBW16Vector.operator[]	574
EBW16Vector.operator=	574
EBW16Vector.RawDataPtr	575
EBW16Vector.SetElement	575
EBW16Vector.WeightedMoment	576
4.22. EBW32PixelAccessor Class	577
EBW32PixelAccessor.EBW32PixelAccessor	577
EBW32PixelAccessor.GetPixel	578
EBW32PixelAccessor.SetPixel	578
4.23. EBW32Vector Class	579
EBW32Vector.AddElement	580
EBW32Vector.Draw	581
EBW32Vector.DrawWithCurrentPen	582
EBW32Vector.EBW32Vector	583
EBW32Vector.GetElement	584
EBW32Vector.operator[]	585
EBW32Vector.operator=	585
EBW32Vector.RawDataPtr	586
EBW32Vector.SetElement	586
EBW32Vector.WeightedMoment	587
4.24. EBW8PathVector Class	587
EBW8PathVector.AddElement	589

EBW8PathVector.Closed	589
EBW8PathVector.Draw	589
EBW8PathVector.DrawWithCurrentPen	591
EBW8PathVector.EBW8PathVector	592
EBW8PathVector.GetElement	592
EBW8PathVector.operator[]	593
EBW8PathVector.operator=	593
EBW8PathVector.RawDataPtr	594
EBW8PathVector.SetElement	594
4.25. EBW8PixelAccessor Class	595
EBW8PixelAccessor.EBW8PixelAccessor	596
EBW8PixelAccessor.GetPixel	596
EBW8PixelAccessor.SetPixel	597
4.26. EBW8Vector Class	597
EBW8Vector.AddElement	599
EBW8Vector.Draw	599
EBW8Vector.DrawWithCurrentPen	601
EBW8Vector.EBW8Vector	602
EBW8Vector.GetElement	602
EBW8Vector.operator[]	603
EBW8Vector.operator=	603
EBW8Vector.RawDataPtr	604
EBW8Vector.SetElement	604
EBW8Vector.WeightedMoment	605
4.27. EBWHistogramVector Class	606
EBWHistogramVector.AddElement	607
EBWHistogramVector.Draw	608
EBWHistogramVector.DrawWithCurrentPen	609
EBWHistogramVector.EBWHistogramVector	610
EBWHistogramVector.GetElement	611
EBWHistogramVector.operator[]	611
EBWHistogramVector.operator=	612
EBWHistogramVector.RawDataPtr	612
EBWHistogramVector.SetElement	612
4.28. EC15PixelAccessor Class	613
EC15PixelAccessor.EC15PixelAccessor	614
EC15PixelAccessor.GetPixel	614
EC15PixelAccessor.SetPixel	615
4.29. EC16PixelAccessor Class	615
EC16PixelAccessor.EC16PixelAccessor	616
EC16PixelAccessor.GetPixel	617
EC16PixelAccessor.SetPixel	617
4.30. EC24APixelAccessor Class	618
EC24APixelAccessor.EC24APixelAccessor	618
EC24APixelAccessor.GetPixel	619
EC24APixelAccessor.SetPixel	620
4.31. EC24PathVector Class	620
EC24PathVector.AddElement	622
EC24PathVector.Closed	622
EC24PathVector.Draw	622
EC24PathVector.DrawWithCurrentPen	624
EC24PathVector.EC24PathVector	625
EC24PathVector.GetElement	625
EC24PathVector.operator[]	626

EC24PathVector.operator=	626
EC24PathVector.RawDataPtr	627
EC24PathVector.SetElement	627
4.32. EC24PixelAccessor Class	628
EC24PixelAccessor.EC24PixelAccessor	629
EC24PixelAccessor.GetPixel	629
EC24PixelAccessor.SetPixel	630
4.33. EC24Vector Class	630
EC24Vector.AddElement	632
EC24Vector.Draw	632
EC24Vector.EC24Vector	635
EC24Vector.GetElement	636
EC24Vector.operator[]	636
EC24Vector.operator=	637
EC24Vector.RawDataPtr	637
EC24Vector.SetElement	637
4.34. ECalibrationGenerator Class	638
4.35. ECalibrationModel Class	638
ECalibrationModel.Apply	639
ECalibrationModel.Create	640
ECalibrationModel.Save	640
ECalibrationModel.Type	641
4.36. ECannyEdgeDetector Class	641
ECannyEdgeDetector.Apply	642
ECannyEdgeDetector.ECannyEdgeDetector	643
ECannyEdgeDetector.HighThreshold	643
ECannyEdgeDetector.LowThreshold	644
ECannyEdgeDetector.ResetSmoothingScale	644
ECannyEdgeDetector.SmoothingScale	645
ECannyEdgeDetector.ThresholdingMode	645
4.37. EChecker Class	646
EChecker.AddPathName	649
EChecker.Attach	650
EChecker.Average	650
EChecker.BatchLearn	651
EChecker.DegreesOfFreedom	651
EChecker.Deviation	652
EChecker.Drag	652
EChecker.Draw	653
EChecker.DrawWithCurrentPen	654
EChecker.EChecker	655
EChecker.EmptyPathNames	656
EChecker.High	656
EChecker.HitHandle	657
EChecker.HitRoi	657
EChecker.HitTest	658
EChecker.Learn	658
EChecker.Load	659
EChecker.Low	660
EChecker.Normalize	660
EChecker.NumAverageSamples	660
EChecker.NumDeviationSamples	661
EChecker.PanX	661
EChecker.PanY	662

EChecker.Register	662
EChecker.Registered	662
EChecker.RelativeTolerance	663
EChecker.Save	663
EChecker.SetPan	664
EChecker.SetTolerance	664
EChecker.SetZoom	665
EChecker.ToleranceX	666
EChecker.ToleranceY	666
EChecker.ZoomX	666
EChecker.ZoomY	667
4.38. ECircle Class	667
ECircle.Amplitude	670
ECircle.Apex	671
ECircle.ApexAngle	671
ECircle.ArcLength	672
ECircle.CopyTo	672
ECircle.Diameter	673
ECircle.Direct	673
ECircle.Distance	674
ECircle.ECircle	674
ECircle.End	676
ECircle.EndAngle	676
ECircle.Full	677
ECircle.GetDistanceBetweenLineAndCircle	677
ECircle.GetDistanceBetweenPointAndCircle	678
ECircle.GetIntersectionOfCircles	679
ECircle.GetIntersectionOfLineAndCircle	680
ECircle.GetPoint	681
ECircle.GetProjectionOfPointOnCircle	681
ECircle.operator=	682
ECircle.Org	682
ECircle.OrgAngle	683
ECircle.Radius	683
ECircle.SetFromCenterAndOrigin	684
ECircle.SetFromOriginMiddleEnd	684
4.39. ECircleGauge Class	685
ECircleGauge.Active	691
ECircleGauge.AddSkipRange	691
ECircleGauge.AverageDistance	692
ECircleGauge.Circle	692
ECircleGauge.CopyTo	693
ECircleGauge.DisableInnerFiltering	693
ECircleGauge.Drag	694
ECircleGauge.Draw	694
ECircleGauge.DrawWithCurrentPen	695
ECircleGauge.ECircleGauge	696
ECircleGauge.FilteringThreshold	697
ECircleGauge.GetMeasuredPeak	697
ECircleGauge.GetMeasuredPoint	698
ECircleGauge.GetMinNumFitSamples	699
ECircleGauge.GetSample	700
ECircleGauge.GetSkipRange	700
ECircleGauge.HitTest	701
ECircleGauge.HVConstraint	702

ECircleGauge.InnerFilteringEnabled	702
ECircleGauge.InnerFilteringThreshold	703
ECircleGauge.Measure	703
ECircleGauge.MeasuredCircle	704
ECircleGauge.MeasureSample	704
ECircleGauge.MeasureWithoutFitting	705
ECircleGauge.MinAmplitude	705
ECircleGauge.MinArea	706
ECircleGauge.NumFilteringPasses	706
ECircleGauge.NumMeasuredPoints	707
ECircleGauge.NumSamples	707
ECircleGauge.NumSkipRanges	708
ECircleGauge.NumValidSamples	708
ECircleGauge.operator=	709
ECircleGauge.Plot	709
ECircleGauge.PlotWithCurrentPen	711
ECircleGauge.Process	712
ECircleGauge.RectangularSamplingArea	712
ECircleGauge.RemoveAllSkipRanges	713
ECircleGauge.RemoveSkipRange	713
ECircleGauge.SamplingStep	713
ECircleGauge.SetMinNumFitSamples	714
ECircleGauge.Smoothing	715
ECircleGauge.Thickness	715
ECircleGauge.Threshold	716
ECircleGauge.Tolerance	716
ECircleGauge.TransitionChoice	717
ECircleGauge.TransitionIndex	717
ECircleGauge.TransitionType	717
ECircleGauge.Type	718
ECircleGauge.Valid	718
4.40. ECircleRegion Class	719
ECircleRegion.Center	720
ECircleRegion.Drag	721
ECircleRegion.ECircleRegion	722
ECircleRegion.HitTest	723
ECircleRegion.Load	724
ECircleRegion.operator!=	725
ECircleRegion.operator=	725
ECircleRegion.operator==	726
ECircleRegion.Radius	726
ECircleRegion.Save	727
ECircleRegion.Scale	727
ECircleRegion.Translate	728
4.41. ECircleShape Class	729
ECircleShape.Amplitude	732
ECircleShape.Angle	733
ECircleShape.Apex	733
ECircleShape.ApexAngle	733
ECircleShape.ArcLength	734
ECircleShape.Center	734
ECircleShape.CenterX	735
ECircleShape.CenterY	735
ECircleShape.Circle	736
ECircleShape.Closest	736

ECircleShape.CopyTo	736
ECircleShape.Diameter	737
ECircleShape.Direct	737
ECircleShape.Drag	738
ECircleShape.Draw	738
ECircleShape.DrawWithCurrentPen	739
ECircleShape.End	740
ECircleShape.EndAngle	740
ECircleShape.Full	741
ECircleShape.GetPoint	741
ECircleShape.HitTest	742
ECircleShape.operator=	742
ECircleShape.Org	743
ECircleShape.OrgAngle	743
ECircleShape.Radius	744
ECircleShape.Scale	744
ECircleShape.SetCenterXY	745
ECircleShape.SetFromCenterAndOrigin	745
ECircleShape.SetFromOriginMiddleEnd	746
ECircleShape.Type	747
4.42. EClassificationDataset Class	747
EClassificationDataset.AddImage	754
EClassificationDataset.AddImages	755
EClassificationDataset.Channels	756
EClassificationDataset.Clear	756
EClassificationDataset.EClassificationDataset	757
EClassificationDataset.EnableDataAugmentation	757
EClassificationDataset.EnableHorizontalFlip	758
EClassificationDataset.EnableVerticalFlip	758
EClassificationDataset.Export	758
EClassificationDataset.GaussianNoiseMaximumStandardDeviation	759
EClassificationDataset.GaussianNoiseMinimumStandardDeviation	760
EClassificationDataset.GetImage	760
EClassificationDataset.GetImageLabel	761
EClassificationDataset.GetImagePath	761
EClassificationDataset.GetImages	762
EClassificationDataset.GetImagesIndexesWithLabel	762
EClassificationDataset.GetLabel	763
EClassificationDataset.GetLabelWeight	763
EClassificationDataset.Height	764
EClassificationDataset.Load	764
EClassificationDataset.MaxBrightnessOffset	765
EClassificationDataset.MaxContrastGain	765
EClassificationDataset.MaxGamma	766
EClassificationDataset.MaxHorizontalShear	766
EClassificationDataset.MaxHorizontalShift	767
EClassificationDataset.MaxHueOffset	767
EClassificationDataset.MaxRotationAngle	768
EClassificationDataset.MaxSaturationGain	768
EClassificationDataset.MaxScale	769
EClassificationDataset.MaxVerticalShear	769
EClassificationDataset.MaxVerticalShift	769
EClassificationDataset.MinContrastGain	770
EClassificationDataset.MinGamma	770
EClassificationDataset.MinSaturationGain	771

EClassificationDataset.MinScale	771
EClassificationDataset.NumImages	772
EClassificationDataset.NumLabels	772
EClassificationDataset.operator=	773
EClassificationDataset.SaltAndPepperNoiseMaximumDensity	773
EClassificationDataset.SaltAndPepperNoiseMinimumDensity	774
EClassificationDataset.Save	774
EClassificationDataset.Serialize	775
EClassificationDataset.SetImageLabel	775
EClassificationDataset.SetLabel	776
EClassificationDataset.SetLabelWeight	777
EClassificationDataset.SpeckleNoiseMaximumStandardDeviation	777
EClassificationDataset.SpeckleNoiseMinimumStandardDeviation	778
EClassificationDataset.SplitDataset	778
EClassificationDataset.Width	779
4.43. EClassificationMetrics Class	780
EClassificationMetrics.Accuracy	781
EClassificationMetrics.AddMetrics	782
EClassificationMetrics.AddResult	782
EClassificationMetrics.EClassificationMetrics	783
EClassificationMetrics.Error	784
EClassificationMetrics.GetConfusion	784
EClassificationMetrics.IsValid	785
EClassificationMetrics.Load	785
EClassificationMetrics.operator=	786
EClassificationMetrics.Save	786
EClassificationMetrics.Serialize	787
4.44. EClassificationResult Class	787
EClassificationResult.BestLabel	788
EClassificationResult.BestLabelId	789
EClassificationResult.BestProbability	789
EClassificationResult.EClassificationResult	790
EClassificationResult.GetProbability	790
EClassificationResult.GetRanking	791
EClassificationResult.IsValid	791
EClassificationResult.operator=	792
4.45. EClassifier Class	792
EClassifier.BatchSize	798
EClassifier.BatchSizeForMaximumClassificationSpeed	798
EClassifier.BestIteration	799
EClassifier.Channels	799
EClassifier.Classify	800
EClassifier.CurrentTrainingFinishedIterations	801
EClassifier.CurrentTrainingNumIterations	801
EClassifier.CurrentTrainingProgression	801
EClassifier.EClassifier	802
EClassifier.EnableAutomaticImageReformat	802
EClassifier.EnableGPU	803
EClassifier.EnableHistogramEqualization	803
EClassifier.Evaluate	804
EClassifier.GetHeatMap	804
EClassifier.GetLabel	805
EClassifier.GetTrainingMetrics	805
EClassifier.GetValidationMetrics	806
EClassifier.GPUIndexes	806

EClassifier.Height	807
EClassifier.ImageCacheSize	807
EClassifier.IsTrained	808
EClassifier.IsTraining	808
EClassifier.Load	809
EClassifier.MinimumHeight	809
EClassifier.MinimumWidth	809
EClassifier.NumGPUs	810
EClassifier.NumLabels	810
EClassifier.NumTrainedIterations	811
EClassifier.operator=	811
EClassifier.OptimizeBatchSize	812
EClassifier.Save	812
EClassifier.Serialize	813
EClassifier.StopTraining	813
EClassifier.Train	814
EClassifier.WaitForIterationCompletion	814
EClassifier.WaitForTrainingCompletion	815
EClassifier.Width	815
4.46. ECodedElement Class	816
ECodedElement.Area	824
ECodedElement.AsHole	824
ECodedElement.AsObject	825
ECodedElement.BottomLimit	825
ECodedElement.BoundingBox	826
ECodedElement.BoundingBoxCenter	826
ECodedElement.BoundingBoxCenterX	826
ECodedElement.BoundingBoxCenterY	827
ECodedElement.BoundingBoxHeight	827
ECodedElement.BoundingBoxWidth	828
ECodedElement.ComputeConvexHull	828
ECodedElement.ComputeFeretBox	828
ECodedElement.ComputePixelGrayAverage	829
ECodedElement.ComputePixelGrayDeviation	829
ECodedElement.ComputePixelGrayVariance	830
ECodedElement.ComputePixelMax	830
ECodedElement.ComputePixelMin	831
ECodedElement.ComputeWeightedGravityCenter	831
ECodedElement.Contour	832
ECodedElement.ContourX	832
ECodedElement.ContourY	833
ECodedElement.Eccentricity	833
ECodedElement.ElementIndex	834
ECodedElement.EllipseAngle	834
ECodedElement.EllipseHeight	835
ECodedElement.EllipseWidth	835
ECodedElement.FeretBox22Box	836
ECodedElement.FeretBox22Center	836
ECodedElement.FeretBox22CenterX	836
ECodedElement.FeretBox22CenterY	837
ECodedElement.FeretBox22Height	837
ECodedElement.FeretBox22Width	838
ECodedElement.FeretBox45Box	838
ECodedElement.FeretBox45Center	838
ECodedElement.FeretBox45CenterX	839

ECodedElement.FeretBox45CenterY	839
ECodedElement.FeretBox45Height	840
ECodedElement.FeretBox45Width	840
ECodedElement.FeretBox68Box	840
ECodedElement.FeretBox68Center	841
ECodedElement.FeretBox68CenterX	841
ECodedElement.FeretBox68CenterY	842
ECodedElement.FeretBox68Height	842
ECodedElement.FeretBox68Width	842
ECodedElement.GetCentralMoment	843
ECodedElement.GetMoment	843
ECodedElement.GetNormalizedCentralMoment	844
ECodedElement.GravityCenter	845
ECodedElement.GravityCenterX	845
ECodedElement.GravityCenterY	846
ECodedElement.IsCodedElement	846
ECodedElement.IsHole	847
ECodedElement.IsObject	847
ECodedElement.LargestRun	847
ECodedElement.LayerIndex	848
ECodedElement.LeftLimit	848
ECodedElement.MinimumEnclosingRectangle	849
ECodedElement.MinimumEnclosingRectangleAngle	849
ECodedElement.MinimumEnclosingRectangleCenter	850
ECodedElement.MinimumEnclosingRectangleCenterX	850
ECodedElement.MinimumEnclosingRectangleCenterY	851
ECodedElement.MinimumEnclosingRectangleHeight	851
ECodedElement.MinimumEnclosingRectangleWidth	851
ECodedElement.RenderMask	852
ECodedElement.RightLimit	853
ECodedElement.RunCount	853
ECodedElement.RunsIterator	853
ECodedElement.SigmaX	854
ECodedElement.SigmaXX	854
ECodedElement.SigmaXY	855
ECodedElement.SigmaY	855
ECodedElement.SigmaYY	855
ECodedElement.TopLimit	856
4.47. ECodedImage Class	856
ECodedImage.AddFeat	867
ECodedImage.AnalyseObjects	867
ECodedImage.BlackClass	869
ECodedImage.BlankFeatures	869
ECodedImage.BuildHoles	869
ECodedImage.BuildLabeledObjects	870
ECodedImage.BuildLabeledRuns	871
ECodedImage.BuildObjects	872
ECodedImage.BuildRuns	873
ECodedImage.Connexity	874
ECodedImage.Continuous	874
ECodedImage.CurrentObjPtr	874
ECodedImage.CurrentRunPtr	875
ECodedImage.DrawDiagonals	875
ECodedImage.DrawObject	876
ECodedImage.DrawObjectFeature	878

ECodedImage.DrawObjectFeatureWithCurrentPen	880
ECodedImage.DrawObjects	882
ECodedImage.DrawObjectsFeature	883
ECodedImage.DrawObjectsFeatureWithCurrentPen	885
ECodedImage.DrawObjectsWithCurrentPen	886
ECodedImage.DrawObjectWithCurrentPen	887
ECodedImage.ECodedImage	888
ECodedImage.FeatureAverage	889
ECodedImage.FeatureDeviation	890
ECodedImage.FeatureMaximum	890
ECodedImage.FeatureMinimum	891
ECodedImage.FeatureVariance	892
ECodedImage.FirstObjPtr	892
ECodedImage.GetCurrentObjData	893
ECodedImage.GetCurrentRunData	893
ECodedImage.GetFeatData	894
ECodedImage.GetFeatDataSize	894
ECodedImage.GetFeatDataType	895
ECodedImage.GetFeatNum	896
ECodedImage.GetFeatPtrByNum	896
ECodedImage.GetFeatSize	897
ECodedImage.GetFirstHole	898
ECodedImage.GetFirstObjData	898
ECodedImage.GetFirstRunData	899
ECodedImage.GetFirstRunPtr	899
ECodedImage.GetHoleParentObject	900
ECodedImage.GetLastObjData	900
ECodedImage.GetLastRunData	901
ECodedImage.GetLastRunPtr	901
ECodedImage.GetNextHole	901
ECodedImage.GetNextObjData	902
ECodedImage.GetNextObjPtr	903
ECodedImage.GetNextRunData	903
ECodedImage.GetNextRunPtr	904
ECodedImage.GetNumHoles	904
ECodedImage.GetNumObjectRuns	905
ECodedImage.GetObjDataPtr	905
ECodedImage.GetObjectData	906
ECodedImage.GetObjectFeature	907
ECodedImage.GetObjFirstRunPtr	909
ECodedImage.GetObjLastRunPtr	910
ECodedImage.GetObjPtr	910
ECodedImage.GetObjPtrByCoordinates	911
ECodedImage.GetObjPtrByPos	912
ECodedImage.GetPreviousObjData	912
ECodedImage.GetPreviousObjPtr	913
ECodedImage.GetPreviousRunData	913
ECodedImage.GetPreviousRunPtr	914
ECodedImage.GetRunData	914
ECodedImage.GetRunDataPtr	915
ECodedImage.GetRunPtr	916
ECodedImage.GetRunPtrByCoordinates	916
ECodedImage.HighColorThreshold	917
ECodedImage.HighImage	917
ECodedImage.HighThreshold	918

ECodedImage.IsHole	918
ECodedImage.IsObjectSelected	919
ECodedImage.LastObjPtr	919
ECodedImage.LimitAngle	920
ECodedImage.LowColorThreshold	920
ECodedImage.LowImage	921
ECodedImage.LowThreshold	921
ECodedImage.MaxObjects	922
ECodedImage.NeutralClass	922
ECodedImage.NumFeatures	923
ECodedImage.NumHoleRuns	923
ECodedImage.NumObjects	923
ECodedImage.NumRuns	924
ECodedImage.NumSelectedObjects	924
ECodedImage.ObjectConvexHull	925
ECodedImage.RemoveAllFeats	925
ECodedImage.RemoveAllObjects	926
ECodedImage.RemoveAllRuns	926
ECodedImage.RemoveHoles	927
ECodedImage.RemoveObject	927
ECodedImage.RemoveRun	928
ECodedImage.ResetContinuousMode	928
ECodedImage.SelectAllObjects	929
ECodedImage.SelectHoles	929
ECodedImage.SelectObject	930
ECodedImage.SelectObjectsUsingFeature	931
ECodedImage.SelectObjectsUsingPosition	932
ECodedImage.SetFeatInfo	933
ECodedImage.SetFirstRunPtr	933
ECodedImage.SetLastRunPtr	934
ECodedImage.SortObjectsUsingFeature	935
ECodedImage.Threshold	935
ECodedImage.ThresholdImage	936
ECodedImage.TrueThreshold	936
ECodedImage.UnselectAllObjects	937
ECodedImage.UnselectHoles	937
ECodedImage.UnselectObject	938
ECodedImage.WhiteClass	938
4.48. ECodedImage2 Class	939
ECodedImage2.ClearFeatureCache	941
ECodedImage2.Draw	942
ECodedImage2.DrawFeature	946
ECodedImage2.DrawFeatureWithCurrentPen	951
ECodedImage2.DrawHole	954
ECodedImage2.DrawHoleFeature	956
ECodedImage2.DrawHoleFeatureWithCurrentPen	959
ECodedImage2.DrawHoleWithCurrentPen	961
ECodedImage2.DrawObject	963
ECodedImage2.DrawObjectFeature	965
ECodedImage2.DrawObjectFeatureWithCurrentPen	968
ECodedImage2.DrawObjectWithCurrentPen	970
ECodedImage2.DrawWithCurrentPen	971
ECodedImage2.ECodedImage2	973
ECodedImage2.FindObject	974
ECodedImage2.GetObj	974

ECodedImage2.GetObjCount	975
ECodedImage2.GetParentObject	976
ECodedImage2.Height	977
ECodedImage2.LayerCount	977
ECodedImage2.RenderMask	978
ECodedImage2.StartY	979
ECodedImage2.Width	979
4.49. EColorLookup Class	980
EColorLookup.AdjustGainOffset	981
EColorLookup.Calibrate	982
EColorLookup.ColorSystemIn	985
EColorLookup.ColorSystemOut	985
EColorLookup.ConvertFromRgb	986
EColorLookup.ConvertToRgb	987
EColorLookup.EColorLookup	987
EColorLookup.IndexBits	988
EColorLookup.Interpolation	988
EColorLookup.Transform	989
EColorLookup.WhiteBalance	990
4.50. EColorRangeThresholdSegmenter Class	991
EColorRangeThresholdSegmenter.HighThreshold	992
EColorRangeThresholdSegmenter.LowThreshold	992
4.51. EColorSingleThresholdSegmenter Class	992
EColorSingleThresholdSegmenter.Threshold	993
4.52. EColorVector Class	993
EColorVector.AddElement	994
EColorVector.EColorVector	995
EColorVector.GetElement	996
EColorVector.operator[]	996
EColorVector.operator=	997
EColorVector.RawDataPtr	997
EColorVector.SetElement	997
4.53. EConverter Class	998
EConverter.Convert	998
4.54. EDecimator Class	1001
EDecimator.Decimate	1001
EDecimator.EDecimator	1002
EDecimator.Load	1003
EDecimator.Save	1003
4.55. EDepthMap Class	1004
EDepthMap.AddMetadata	1007
EDepthMap.AxisSystemType	1008
EDepthMap.Clear	1008
EDepthMap.ClearMetadata	1008
EDepthMap.ConvertCoordinatesMapToPixel	1009
EDepthMap.ConvertCoordinatesPixelToMap	1010
EDepthMap.DeleteMetadata	1010
EDepthMap.Draw	1011
EDepthMap.DrawImage	1014
EDepthMap.GetBufferPtr	1017
EDepthMap.GetCheckedBufferPtr	1018
EDepthMap.GetMetadata	1018
EDepthMap.GetZValue	1019
EDepthMap.Height	1019

EDepthMap.IsVoid	1020
EDepthMap.Load	1020
EDepthMap.LoadImage	1021
EDepthMap.LoadImageAndMetadata	1021
EDepthMap.LoadMetadata	1022
EDepthMap.ModifyMetadata	1022
EDepthMap.RowPitch	1023
EDepthMap.Save	1023
EDepthMap.SaveImage	1024
EDepthMap.SaveImageAndMetadata	1025
EDepthMap.SaveJpeg	1025
EDepthMap.SaveJpeg2K	1026
EDepthMap.SaveMetadata	1027
EDepthMap.Serialize	1027
EDepthMap.SerializeImage	1028
EDepthMap.SetBufferPtr	1028
EDepthMap.SetSize	1029
EDepthMap.Type	1030
EDepthMap.Width	1030
EDepthMap.ZResolution	1030
4.56. EDepthMap16 Class	1031
EDepthMap16.AddMetadata	1035
EDepthMap16.AsEImage	1035
EDepthMap16.AxisSystemType	1036
EDepthMap16.Clear	1036
EDepthMap16.ClearMetadata	1037
EDepthMap16.ConvertCoordinatesMapToPixel	1037
EDepthMap16.ConvertCoordinatesPixelToMap	1038
EDepthMap16.CopyMetadataTo	1038
EDepthMap16.DeleteMetadata	1039
EDepthMap16.Draw	1039
EDepthMap16.DrawImage	1043
EDepthMap16.EDepthMap16	1045
EDepthMap16.FillUndefinedPixels	1046
EDepthMap16.GetBufferPtr	1046
EDepthMap16.GetCheckedBufferPtr	1047
EDepthMap16.GetMetadata	1048
EDepthMap16.GetPixel	1049
EDepthMap16.GetZValue	1049
EDepthMap16.Height	1050
EDepthMap16.IsVoid	1050
EDepthMap16.Load	1051
EDepthMap16.LoadImage	1051
EDepthMap16.LoadImageAndMetadata	1052
EDepthMap16.LoadMetadata	1052
EDepthMap16.ModifyMetadata	1053
EDepthMap16.operator=	1053
EDepthMap16.RowPitch	1054
EDepthMap16.Save	1054
EDepthMap16.SaveImage	1055
EDepthMap16.SaveImageAndMetadata	1056
EDepthMap16.SaveJpeg	1056
EDepthMap16.SaveJpeg2K	1057
EDepthMap16.SaveMetadata	1058
EDepthMap16.Serialize	1058

EDepthMap16.SerializelImage	1059
EDepthMap16.SetBufferPtr	1059
EDepthMap16.SetPixel	1060
EDepthMap16.SetSize	1060
EDepthMap16.Type	1061
EDepthMap16.UndefinedValue	1062
EDepthMap16.Width	1062
EDepthMap16.ZResolution	1062
4.57. EDepthMap32f Class	1063
EDepthMap32f.AddMetadata	1067
EDepthMap32f.AsEImage	1067
EDepthMap32f.AxisSystemType	1068
EDepthMap32f.Clear	1068
EDepthMap32f.ClearMetadata	1069
EDepthMap32f.ConvertCoordinatesMapToPixel	1069
EDepthMap32f.ConvertCoordinatesPixelToMap	1070
EDepthMap32f.CopyMetadataTo	1070
EDepthMap32f.DeleteMetadata	1071
EDepthMap32f.Draw	1071
EDepthMap32f.DrawImage	1075
EDepthMap32f.EDepthMap32f	1077
EDepthMap32f.FillUndefinedPixels	1078
EDepthMap32f.GetBufferPtr	1078
EDepthMap32f.GetCheckedBufferPtr	1079
EDepthMap32f.GetMetadata	1080
EDepthMap32f.GetPixel	1081
EDepthMap32f.GetZValue	1081
EDepthMap32f.Height	1082
EDepthMap32f.IsVoid	1082
EDepthMap32f.Load	1083
EDepthMap32f.LoadImage	1083
EDepthMap32f.LoadImageAndMetadata	1084
EDepthMap32f.LoadMetadata	1084
EDepthMap32f.ModifyMetadata	1085
EDepthMap32f.operator=	1085
EDepthMap32f.RowPitch	1086
EDepthMap32f.Save	1086
EDepthMap32f.SaveImage	1087
EDepthMap32f.SaveImageAndMetadata	1088
EDepthMap32f.SaveJpeg	1088
EDepthMap32f.SaveJpeg2K	1089
EDepthMap32f.SaveMetadata	1090
EDepthMap32f.Serialize	1090
EDepthMap32f.SerializelImage	1091
EDepthMap32f.SetBufferPtr	1091
EDepthMap32f.SetPixel	1092
EDepthMap32f.SetSize	1092
EDepthMap32f.Type	1093
EDepthMap32f.UndefinedValue	1094
EDepthMap32f.Width	1094
EDepthMap32f.ZResolution	1094
4.58. EDepthMap8 Class	1095
EDepthMap8.AddMetadata	1099
EDepthMap8.AsEImage	1099
EDepthMap8.AxisSystemType	1100

EDepthMap8.Clear	1100
EDepthMap8.ClearMetadata	1101
EDepthMap8.ConvertCoordinatesMapToPixel	1101
EDepthMap8.ConvertCoordinatesPixelToMap	1102
EDepthMap8.CopyMetadataTo	1102
EDepthMap8.DeleteMetadata	1103
EDepthMap8.Draw	1103
EDepthMap8.DrawImage	1107
EDepthMap8.EDepthMap8	1109
EDepthMap8.FillUndefinedPixels	1110
EDepthMap8.GetBufferPtr	1110
EDepthMap8.GetCheckedBufferPtr	1111
EDepthMap8.GetMetadata	1112
EDepthMap8.GetPixel	1113
EDepthMap8.GetZValue	1113
EDepthMap8.Height	1114
EDepthMap8.IsVoid	1114
EDepthMap8.Load	1115
EDepthMap8.LoadImage	1115
EDepthMap8.LoadImageAndMetadata	1116
EDepthMap8.LoadMetadata	1116
EDepthMap8.ModifyMetadata	1117
EDepthMap8.operator=	1117
EDepthMap8.RowPitch	1118
EDepthMap8.Save	1118
EDepthMap8.SaveImage	1119
EDepthMap8.SaveImageAndMetadata	1120
EDepthMap8.SaveJpeg	1120
EDepthMap8.SaveJpeg2K	1121
EDepthMap8.SaveMetadata	1122
EDepthMap8.Serialize	1122
EDepthMap8.SerializeImage	1123
EDepthMap8.SetBufferPtr	1123
EDepthMap8.SetPixel	1124
EDepthMap8.SetSize	1124
EDepthMap8.Type	1125
EDepthMap8.UndefinedValue	1126
EDepthMap8.Width	1126
EDepthMap8.ZResolution	1126
4.59. EDepthMapToMeshConverter Class	1127
EDepthMapToMeshConverter.CalibrationModel	1128
EDepthMapToMeshConverter.Convert	1128
EDepthMapToMeshConverter.EDepthMapToMeshConverter	1129
EDepthMapToMeshConverter.Load	1130
EDepthMapToMeshConverter.operator=	1130
EDepthMapToMeshConverter.Save	1131
4.60. EDepthMapToPointCloudConverter Class	1132
EDepthMapToPointCloudConverter.CalibrationModel	1133
EDepthMapToPointCloudConverter.Convert	1133
EDepthMapToPointCloudConverter.EDepthMapToPointCloudConverter	1134
EDepthMapToPointCloudConverter.Load	1135
EDepthMapToPointCloudConverter.operator=	1135
EDepthMapToPointCloudConverter.Save	1136
4.61. EEllipseRegion Class	1136
EEllipseRegion.Angle	1138

EEllipseRegion.Center	1139
EEllipseRegion.Drag	1139
EEllipseRegion.EEllipseRegion	1140
EEllipseRegion.HitTest	1142
EEllipseRegion.HorizontalRadius	1143
EEllipseRegion.Load	1143
EEllipseRegion.operator!=	1144
EEllipseRegion.operator=	1144
EEllipseRegion.operator==	1145
EEllipseRegion.Rotate	1145
EEllipseRegion.Save	1146
EEllipseRegion.Scale	1146
EEllipseRegion.Translate	1147
EEllipseRegion.VerticalRadius	1148
4.62. EErrorStatistics Class	1148
EErrorStatistics.CopyTo	1149
EErrorStatistics.EErrorStatistics	1150
EErrorStatistics.Load	1150
EErrorStatistics.Max	1151
EErrorStatistics.Mean	1151
EErrorStatistics.Min	1152
EErrorStatistics.NumOfErrors	1152
EErrorStatistics.NumOfValidSamples	1152
EErrorStatistics.operator=	1153
EErrorStatistics.Save	1153
EErrorStatistics.StdDev	1154
4.63. EException Class	1154
EException.EException	1155
EException.Error	1156
EException.operator=	1156
EException.What	1157
4.64. EExplicitGeometricCalibrationModel Class	1157
EExplicitGeometricCalibrationModel.CameraAngle	1159
EExplicitGeometricCalibrationModel.CameraHeight	1160
EExplicitGeometricCalibrationModel.EExplicitGeometricCalibrationModel	1160
EExplicitGeometricCalibrationModel.FocalLength	1162
EExplicitGeometricCalibrationModel.IsInitialized	1162
EExplicitGeometricCalibrationModel.LaserPlaneAngle	1163
EExplicitGeometricCalibrationModel.Load	1163
EExplicitGeometricCalibrationModel.MotionIncrement	1164
EExplicitGeometricCalibrationModel.operator=	1164
EExplicitGeometricCalibrationModel.operator==	1164
EExplicitGeometricCalibrationModel.RoiBottomLine	1165
EExplicitGeometricCalibrationModel.RoiLeftColumn	1165
EExplicitGeometricCalibrationModel.Save	1166
EExplicitGeometricCalibrationModel.SensorHeight	1166
EExplicitGeometricCalibrationModel.SensorWidth	1167
EExplicitGeometricCalibrationModel.SensorXResolution	1167
EExplicitGeometricCalibrationModel.SensorYResolution	1167
EExplicitGeometricCalibrationModel.Type	1168
4.65. EFeaturesAligner Class	1168
EFeaturesAligner.Compute	1169
EFeaturesAligner.EFeaturesAligner	1170
EFeaturesAligner.Load	1171
EFeaturesAligner.ModelPoints	1171

EFeaturesAligner.operator=	1172
EFeaturesAligner.PolarityTransform	1172
EFeaturesAligner.Save	1173
4.66. EFilePointerSerializer Class	1173
EFilePointerSerializer.Close	1174
EFilePointerSerializer.Writing	1174
4.67. EFileSerializer Class	1174
EFileSerializer.Close	1175
EFileSerializer.Writing	1175
4.68. EFilters Class	1176
EFilters.RemoveNoise	1176
4.69. EFloatRange Class	1178
EFloatRange.Center	1180
EFloatRange.EFloatRange	1180
EFloatRange.IsInRange	1181
EFloatRange.LowerBound	1181
EFloatRange.operator=	1182
EFloatRange.operator==	1182
EFloatRange.SetBounds	1183
EFloatRange.SetFromBaseAndAbsoluteTolerance	1183
EFloatRange.SetFromBaseAndRelativeTolerance	1184
EFloatRange.Size	1185
EFloatRange.Update	1185
EFloatRange.UpperBound	1185
4.70. EFoundPattern Class	1186
EFoundPattern.Angle	1188
EFoundPattern.Center	1188
EFoundPattern.Draw	1189
EFoundPattern.DrawBoundingBox	1190
EFoundPattern.DrawCenter	1190
EFoundPattern.DrawFeaturePoints	1191
EFoundPattern.DrawWithCurrentPen	1191
EFoundPattern.EFoundPattern	1192
EFoundPattern.operator!=	1193
EFoundPattern.operator=	1193
EFoundPattern.operator==	1194
EFoundPattern.Quadrangle	1194
EFoundPattern.Scale	1195
EFoundPattern.Score	1195
4.71. EFrame Class	1196
EFrame.Angle	1197
EFrame.CenterX	1197
EFrame.CenterY	1198
EFrame.CopyTo	1198
EFrame.EFrame	1199
EFrame.GlobalToLocal	1200
EFrame.LocalToGlobal	1200
EFrame.operator=	1201
EFrame.Scale	1202
4.72. EFrameShape Class	1202
EFrameShape.Angle	1204
EFrameShape.Center	1205
EFrameShape.CenterX	1205
EFrameShape.CenterY	1205

EFrameShape.Closest	1206
EFrameShape.CopyTo	1206
EFrameShape.Drag	1207
EFrameShape.Draw	1207
EFrameShape.DrawWithCurrentPen	1208
EFrameShape.EFrameShape	1209
EFrameShape.HitTest	1210
EFrameShape.operator=	1210
EFrameShape.Scale	1211
EFrameShape.Set	1211
EFrameShape.SetCenterXY	1212
EFrameShape.SetSize	1212
EFrameShape.SizeX	1213
EFrameShape.SizeY	1214
EFrameShape.Type	1214
4.73. EGrabberDepthMap16 Class	1215
EGrabberDepthMap16.EGrabberDepthMap16	1215
4.74. EGrabberDepthMap8 Class	1216
EGrabberDepthMap8.EGrabberDepthMap8	1216
4.75. EGrabberImageBW16 Class	1217
EGrabberImageBW16.EGrabberImageBW16	1218
4.76. EGrabberImageBW8 Class	1219
EGrabberImageBW8.EGrabberImageBW8	1219
4.77. EGrabberImageC24 Class	1220
EGrabberImageC24.EGrabberImageC24	1220
4.78. EGrayscaleDoubleThresholdSegmenter Class	1221
EGrayscaleDoubleThresholdSegmenter.HighThreshold	1222
EGrayscaleDoubleThresholdSegmenter.LowThreshold	1222
4.79. EGrayscaleSingleThresholdSegmenter Class	1223
EGrayscaleSingleThresholdSegmenter.AbsoluteThreshold	1224
EGrayscaleSingleThresholdSegmenter.IsFirstApplication	1224
EGrayscaleSingleThresholdSegmenter.LastThreshold	1225
EGrayscaleSingleThresholdSegmenter.Mode	1225
EGrayscaleSingleThresholdSegmenter.RelativeThreshold	1225
4.80. EHarrisCornerDetector Class	1226
EHarrisCornerDetector.Apply	1227
EHarrisCornerDetector.DerivationScale	1227
EHarrisCornerDetector.EHarrisCornerDetector	1228
EHarrisCornerDetector.GradientNormalizationEnabled	1228
EHarrisCornerDetector.IntegrationScale	1229
EHarrisCornerDetector.SubpixelPrecisionEnabled	1229
EHarrisCornerDetector.Threshold	1230
EHarrisCornerDetector.ThresholdingMode	1230
4.81. EHarrisInterestPoints Class	1231
EHarrisInterestPoints.Draw	1232
EHarrisInterestPoints.DrawCorner	1234
EHarrisInterestPoints.DrawCornerWithCurrentPen	1235
EHarrisInterestPoints.DrawWithCurrentPen	1236
EHarrisInterestPoints.EHarrisInterestPoints	1237
EHarrisInterestPoints.GetCornerness	1237
EHarrisInterestPoints.GetGradientMagnitude	1238
EHarrisInterestPoints.GetGradientOrientation	1238
EHarrisInterestPoints.GetGradientX	1239
EHarrisInterestPoints.GetGradientY	1239

EHarrisInterestPoints.GetPoint	1240
EHarrisInterestPoints.GetX	1240
EHarrisInterestPoints.GetY	1241
EHarrisInterestPoints.PointCount	1241
4.82. EHDRColorFuser Class	1242
EHDRColorFuser.EHDRColorFuser	1243
EHDRColorFuser.Fuse	1243
EHDRColorFuser.GetFusedImage	1244
EHDRColorFuser.operator=	1245
4.83. EHDRFuser Class	1245
EHDRFuser.EHDRFuser	1246
EHDRFuser.Fuse	1247
EHDRFuser.GetFusedImage	1247
EHDRFuser.operator=	1248
4.84. EHitAndMissKernel Class	1249
EHitAndMissKernel.EHitAndMissKernel	1250
EHitAndMissKernel.EndX	1251
EHitAndMissKernel.EndY	1251
EHitAndMissKernel.GetValue	1252
EHitAndMissKernel.SetSize	1252
EHitAndMissKernel.SetValue	1253
EHitAndMissKernel.StartX	1254
EHitAndMissKernel.StartY	1254
4.85. EHole Class	1255
EHole.ParentObjectIndex	1255
4.86. ElmageBW1 Class	1256
ElmageBW1.ElmageBW1	1256
ElmageBW1.GetBitIndex	1257
ElmageBW1.operator=	1258
4.87. ElmageBW16 Class	1259
ElmageBW16.ElmageBW16	1259
ElmageBW16.operator=	1260
4.88. ElmageBW32 Class	1261
ElmageBW32.ElmageBW32	1261
ElmageBW32.operator=	1262
4.89. ElmageBW8 Class	1263
ElmageBW8.ElmageBW8	1263
ElmageBW8.operator=	1264
4.90. ElmageC15 Class	1265
ElmageC15.ElmageC15	1265
ElmageC15.operator=	1266
4.91. ElmageC16 Class	1267
ElmageC16.ElmageC16	1267
ElmageC16.operator=	1268
4.92. ElmageC24 Class	1269
ElmageC24.ElmageC24	1269
ElmageC24.operator=	1270
4.93. ElmageC24A Class	1271
ElmageC24A.ElmageC24A	1271
ElmageC24A.operator=	1272
4.94. ElmageC48 Class	1273
ElmageC48.ElmageC48	1273
ElmageC48.operator=	1274

4.95. ElmageEncoder Class	1275
ElmageEncoder.BinaryImageSegmenter	1277
ElmageEncoder.ColorRangeThresholdSegmenter	1277
ElmageEncoder.ColorSingleThresholdSegmenter	1278
ElmageEncoder.ContinuousModeEnabled	1278
ElmageEncoder.ContinuousModeMaxHeight	1279
ElmageEncoder.ElmageEncoder	1279
ElmageEncoder.Encode	1280
ElmageEncoder.EncodingConnexity	1282
ElmageEncoder.FlushContinuousMode	1282
ElmageEncoder.GrayscaleDoubleThresholdSegmenter	1283
ElmageEncoder.GrayscaleSingleThresholdSegmenter	1283
ElmageEncoder.ImageRangeSegmenter	1284
ElmageEncoder.LabeledImageSegmenter	1284
ElmageEncoder.ReferenceImageSegmenter	1285
ElmageEncoder.ResetContinuousMode	1285
ElmageEncoder.SegmentationMethod	1285
4.96. ElmageRangeSegmenter Class	1286
ElmageRangeSegmenter.BlackLayerEncoded	1287
ElmageRangeSegmenter.BlackLayerIndex	1288
ElmageRangeSegmenter.HighImageBW16	1288
ElmageRangeSegmenter.HighImageBW8	1288
ElmageRangeSegmenter.HighImageC24	1289
ElmageRangeSegmenter.LowImageBW16	1289
ElmageRangeSegmenter.LowImageBW8	1290
ElmageRangeSegmenter.LowImageC24	1290
ElmageRangeSegmenter.WhiteLayerEncoded	1290
ElmageRangeSegmenter.WhiteLayerIndex	1291
4.97. ElmageSegmenter Class	1291
4.98. ElntegerRange Class	1292
ElntegerRange.Center	1293
ElntegerRange.ElntegerRange	1293
ElntegerRange.IsInRange	1294
ElntegerRange.LowerBound	1295
ElntegerRange.operator=	1295
ElntegerRange.SetBounds	1296
ElntegerRange.SetFromBaseAndAbsoluteTolerance	1296
ElntegerRange.SetFromBaseAndRelativeTolerance	1297
ElntegerRange.Size	1298
ElntegerRange.Update	1298
ElntegerRange.UpperBound	1299
4.99. EKernel Class	1299
EKernel.EKernel	1300
EKernel.Gain	1302
EKernel.GetKernelData	1302
EKernel.Offset	1303
EKernel.OutsideValue	1303
EKernel.RawDataPtr	1304
EKernel.Rectifier	1304
EKernel.SetKernelData	1305
EKernel.SetSize	1311
EKernel.SizeX	1311
EKernel.SizeY	1312
4.100. ELabeledImageSegmenter Class	1312

ELabeledImageSegmenter.MaxLayer	1313
ELabeledImageSegmenter.MinLayer	1313
4.101. ELandmark Class	1314
ELandmark.ELandmark	1315
ELandmark.operator=	1315
ELandmark.SensorX	1316
ELandmark.SensorY	1316
ELandmark.WorldX	1316
ELandmark.WorldY	1317
4.102. ELaserLineExtractor Class	1317
ELaserLineExtractor.AnalysisMode	1318
ELaserLineExtractor.AnalysisThreshold	1319
ELaserLineExtractor.DepthMap	1319
ELaserLineExtractor.ELaserLineExtractor	1320
ELaserLineExtractor.EnableSmoothing	1321
ELaserLineExtractor.ExtractProfileFromFrame	1321
ELaserLineExtractor.operator=	1322
ELaserLineExtractor.Profile	1322
ELaserLineExtractor.SetSmoothingParameters	1323
4.103. ELine Class	1323
ELine.CopyTo	1325
ELine.ELine	1326
ELine.End	1327
ELine.GetAngleBetweenLines	1327
ELine.GetDistanceBetweenPointAndLine	1328
ELine.GetIntersectionOfLines	1328
ELine.GetPoint	1329
ELine.GetProjectionOfPointOnLine	1330
ELine.Length	1330
ELine.operator=	1331
ELine.Org	1331
ELine.SetFromOriginAndEnd	1331
ELine.SetFromTwoPoints	1332
4.104. ELineGauge Class	1333
ELineGauge.Active	1338
ELineGauge.AddSkipRange	1339
ELineGauge.AverageDistance	1339
ELineGauge.ClippingMode	1340
ELineGauge.CopyTo	1340
ELineGauge.Drag	1341
ELineGauge.Draw	1342
ELineGauge.DrawWithCurrentPen	1343
ELineGauge.ELineGauge	1343
ELineGauge.FilteringThreshold	1344
ELineGauge.GetMeasuredPeak	1345
ELineGauge.GetMeasuredPoint	1345
ELineGauge.GetMinNumFitSamples	1346
ELineGauge.GetSample	1347
ELineGauge.GetSkipRange	1348
ELineGauge.HitTest	1348
ELineGauge.HVConstraint	1349
ELineGauge.KnownAngle	1349
ELineGauge.Line	1350
ELineGauge.Measure	1350
ELineGauge.MeasuredLine	1351

ELineGauge.MeasureSample	1351
ELineGauge.MeasureWithoutFitting	1352
ELineGauge.MinAmplitude	1353
ELineGauge.MinArea	1353
ELineGauge.NumFilteringPasses	1354
ELineGauge.NumMeasuredPoints	1354
ELineGauge.NumSamples	1355
ELineGauge.NumSkipRanges	1355
ELineGauge.NumValidSamples	1355
ELineGauge.operator=	1356
ELineGauge.Plot	1356
ELineGauge.PlotWithCurrentPen	1358
ELineGauge.Process	1359
ELineGauge.RectangularSamplingArea	1359
ELineGauge.RemoveAllSkipRanges	1360
ELineGauge.RemoveSkipRange	1360
ELineGauge.SamplingStep	1361
ELineGauge.SetMinNumFitSamples	1361
ELineGauge.Smoothing	1362
ELineGauge.Thickness	1363
ELineGauge.Threshold	1363
ELineGauge.Tolerance	1364
ELineGauge.TransitionChoice	1364
ELineGauge.TransitionIndex	1365
ELineGauge.TransitionType	1365
ELineGauge.Type	1366
ELineGauge.Valid	1366
4.105. ELineStyle Class	1367
ELineStyle.Angle	1369
ELineStyle.Center	1370
ELineStyle.CenterX	1370
ELineStyle.CenterY	1370
ELineStyle.Closest	1371
ELineStyle.CopyTo	1371
ELineStyle.Drag	1372
ELineStyle.Draw	1373
ELineStyle.DrawWithCurrentPen	1374
ELineStyle.End	1374
ELineStyle.GetPoint	1375
ELineStyle.HitTest	1375
ELineStyle.Length	1376
ELineStyle.Line	1376
ELineStyle.operator=	1376
ELineStyle.Org	1377
ELineStyle.Scale	1377
ELineStyle.SetCenterXY	1378
ELineStyle.SetFromOriginAndEnd	1378
ELineStyle.SetFromTwoPoints	1379
ELineStyle.Type	1380
4.106. EListItem Class	1380
4.107. EMailBarcode Class	1381
EMailBarcode.ChecksumOk	1382
EMailBarcode.ComponentStrings	1382
EMailBarcode.Draw	1383
EMailBarcode.EMailBarcode	1384

EMailBarcode.operator=	1384
EMailBarcode.Orientation	1385
EMailBarcode.Position	1385
EMailBarcode.Symbology	1385
EMailBarcode.Text	1386
4.108. EMailBarcodeReader Class	1386
EMailBarcodeReader.EMailBarcodeReader	1387
EMailBarcodeReader.EnableClutteredBarcodes	1388
EMailBarcodeReader.EnableDottedBarcodes	1388
EMailBarcodeReader.ExpectedOrientations	1389
EMailBarcodeReader.ExpectedSymbologies	1389
EMailBarcodeReader.Load	1390
EMailBarcodeReader.operator=	1390
EMailBarcodeReader.Read	1391
EMailBarcodeReader.Save	1391
EMailBarcodeReader.ValidateChecksum	1392
4.109. EMatcher Class	1392
EMatcher.AdvancedLearning	1398
EMatcher.AngleStep	1398
EMatcher.ClearImage	1399
EMatcher.ContrastMode	1399
EMatcher.CopyLearntPattern	1400
EMatcher.CopyTo	1400
EMatcher.CorrelationMode	1401
EMatcher.DontCareThreshold	1401
EMatcher.DrawPosition	1402
EMatcher.DrawPositions	1403
EMatcher.DrawPositionsWithCurrentPen	1405
EMatcher.DrawPositionWithCurrentPen	1406
EMatcher.EMatcher	1407
EMatcher.FilteringMode	1408
EMatcher.FinalReduction	1408
EMatcher.GetPixelDimensions	1409
EMatcher.GetPosition	1410
EMatcher.InitialMinScore	1410
EMatcher.Interpolate	1411
EMatcher.IsotropicScale	1411
EMatcher.LearnPattern	1412
EMatcher.Load	1413
EMatcher.Match	1413
EMatcher.MaxAngle	1414
EMatcher.MaxInitialPositions	1414
EMatcher.MaxPositions	1415
EMatcher.MaxScale	1415
EMatcher.MaxScaleX	1416
EMatcher.MaxScaleY	1416
EMatcher.MinAngle	1417
EMatcher.MinReducedArea	1417
EMatcher.MinScale	1418
EMatcher.MinScaleX	1418
EMatcher.MinScaleY	1419
EMatcher.MinScore	1419
EMatcher.NumPositions	1420
EMatcher.NumReductions	1420
EMatcher.operator=	1421

EMatcher.PatternHeight	1421
EMatcher.PatternLearnt	1422
EMatcher.PatternType	1422
EMatcher.PatternWidth	1423
EMatcher.Positions	1423
EMatcher.Save	1423
EMatcher.ScaleStep	1424
EMatcher.ScaleXStep	1424
EMatcher.ScaleYStep	1425
EMatcher.SetExtension	1425
EMatcher.SetPixelDimensions	1426
EMatcher.Version	1426
4.110. EMatrixCode Class	1427
EMatrixCode.Angle	1432
EMatrixCode.AxialNonUniformity	1433
EMatrixCode.AxialNonUniformityGrade	1433
EMatrixCode.CellDefects	1434
EMatrixCode.Center	1434
EMatrixCode.Contrast	1434
EMatrixCode.ContrastGrade	1435
EMatrixCode.ContrastType	1435
EMatrixCode.DataMatrixCellHeight	1436
EMatrixCode.DataMatrixCellWidth	1436
EMatrixCode.DecodedString	1437
EMatrixCode.Draw	1437
EMatrixCode.DrawErrors	1438
EMatrixCode.DrawErrorsWithCurrentPen	1440
EMatrixCode.DrawWithCurrentPen	1440
EMatrixCode.EMatrixCode	1441
EMatrixCode.Family	1442
EMatrixCode.FinderPatternDefects	1442
EMatrixCode.Flipping	1443
EMatrixCode.Found	1443
EMatrixCode.GetCorner	1444
EMatrixCode.GetDecodedDataElement	1444
EMatrixCode.HorizontalMarkGrowth	1445
EMatrixCode.HorizontalMarkMisplacement	1445
EMatrixCode.IsGS1	1446
EMatrixCode.Iso15415GradingParameters	1446
EMatrixCode.Iso29158GradingParameters	1447
EMatrixCode.Load	1447
EMatrixCode.LocationThreshold	1448
EMatrixCode.LogicalSize	1448
EMatrixCode.LogicalSizeHeight	1448
EMatrixCode.LogicalSizeWidth	1449
EMatrixCode.MeasuredPrintGrowth	1449
EMatrixCode.NumErrors	1450
EMatrixCode.operator=	1450
EMatrixCode.OverallGrade	1451
EMatrixCode.PrintGrowth	1451
EMatrixCode.PrintGrowthGrade	1452
EMatrixCode.ReadingThreshold	1452
EMatrixCode.Save	1453
EMatrixCode.SemiT10GradingParameters	1453
EMatrixCode.SetCorner	1454

EMatrixCode.SymbolContrastSNR	1454
EMatrixCode.UnusedErrorCorrection	1455
EMatrixCode.UnusedErrorCorrectionGrade	1455
EMatrixCode.VerticalMarkGrowth	1456
EMatrixCode.VerticalMarkMisplacement	1456
4.111. EMatrixCode Class	1457
EMatrixCode.DecodedString	1459
EMatrixCode.DrawErrors	1459
EMatrixCode.DrawGrid	1460
EMatrixCode.DrawPosition	1461
EMatrixCode.ECC000Family	1461
EMatrixCode.EMatrixCode	1462
EMatrixCode.Errors	1462
EMatrixCode.GetCellPosition	1463
EMatrixCode.IsECC200	1463
EMatrixCode.IsGS1	1464
EMatrixCode.Iso15415GradingParameters	1464
EMatrixCode.Iso29158GradingParameters	1465
EMatrixCode.operator=	1465
EMatrixCode.Position	1465
EMatrixCode.SemiT10GradingParameters	1466
EMatrixCode.SymbolHeight	1466
EMatrixCode.SymbolWidth	1467
4.112. EMatrixCodeReader Class	1467
EMatrixCodeReader.ComputeGrading	1469
EMatrixCodeReader.EMatrixCodeReader	1470
EMatrixCodeReader.GetLearnMaskElement	1470
EMatrixCodeReader.Learn	1471
EMatrixCodeReader.LearnMore	1472
EMatrixCodeReader.Load	1472
EMatrixCodeReader.MaxHeightWidthRatio	1473
EMatrixCodeReader.MaximumPrintGrowth	1473
EMatrixCodeReader.MinimumPrintGrowth	1474
EMatrixCodeReader.NominalPrintGrowth	1474
EMatrixCodeReader.Read	1475
EMatrixCodeReader.Reset	1476
EMatrixCodeReader.Save	1476
EMatrixCodeReader.SearchParams	1477
EMatrixCodeReader.SetIso29158CalibrationParameters	1477
EMatrixCodeReader.SetLearnMaskElement	1478
EMatrixCodeReader.TimeOut	1478
4.113. EMatrixCodeReader Class	1479
EMatrixCodeReader.ComputeGrading	1481
EMatrixCodeReader.EMatrixCodeReader	1481
EMatrixCodeReader.Iso29158CalibrationParameters	1482
EMatrixCodeReader.Learn	1482
EMatrixCodeReader.Load	1483
EMatrixCodeReader.MaxNumCodes	1483
EMatrixCodeReader.operator=	1484
EMatrixCodeReader.Read	1484
EMatrixCodeReader.ReadMode	1485
EMatrixCodeReader.ReadResults	1485
EMatrixCodeReader.ResetLearning	1485
EMatrixCodeReader.Save	1486
EMatrixCodeReader.StopProcess	1486

EMatrixCodeReader.TimeOut	1487
4.114. EMeasurementUnit Class	1487
EMeasurementUnit.ConversionFactorTo	1488
EMeasurementUnit.EMeasurementUnit	1489
EMeasurementUnit.GetStockMeasurementUnit	1490
EMeasurementUnit.Magnitude	1490
EMeasurementUnit.Name	1490
4.115. EMemorySerializer Class	1491
EMemorySerializer.Buffer	1492
EMemorySerializer.BufferSize	1492
EMemorySerializer.Close	1492
EMemorySerializer.CurrentPosition	1493
EMemorySerializer.Writing	1493
4.116. EMesh Class	1494
EMesh.EMesh	1495
EMesh.Load	1496
EMesh.LoadSTL	1497
EMesh.operator=	1497
EMesh.PointCloud	1498
EMesh.Save	1498
EMesh.SaveSTL	1499
EMesh.TriangleCount	1499
EMesh.TriangleIndexes	1500
4.117. EMeshToZMapConverter Class	1500
EMeshToZMapConverter.Convert	1505
EMeshToZMapConverter.EMeshToZMapConverter	1506
EMeshToZMapConverter.EnableFillMode	1507
EMeshToZMapConverter.Extension	1507
EMeshToZMapConverter.FillUndefinedPixelsDirection	1508
EMeshToZMapConverter.FillUndefinedPixelsMethod	1508
EMeshToZMapConverter.IsFillModeEnabled	1508
EMeshToZMapConverter.Load	1509
EMeshToZMapConverter.MapHeight	1509
EMeshToZMapConverter.MapWidth	1510
EMeshToZMapConverter.MapXResolution	1510
EMeshToZMapConverter.MapYResolution	1511
EMeshToZMapConverter.MapZResolution	1511
EMeshToZMapConverter.operator=	1511
EMeshToZMapConverter.operator==	1512
EMeshToZMapConverter.OrientationVector	1512
EMeshToZMapConverter.OrientationVectorMode	1513
EMeshToZMapConverter.Origin	1513
EMeshToZMapConverter.ReferencePlane	1514
EMeshToZMapConverter.ReferencePlaneMode	1514
EMeshToZMapConverter.Save	1515
EMeshToZMapConverter.SetFillMode	1515
EMeshToZMapConverter.SetMapSize	1516
EMeshToZMapConverter.SetMapXYResolution	1517
EMeshToZMapConverter.UnsetMapSize	1517
EMeshToZMapConverter.UnsetMapXYResolution	1518
EMeshToZMapConverter.UnsetMapZResolution	1518
EMeshToZMapConverter.UnsetOrigin	1519
EMeshToZMapConverter.UnsetWorldToZMapTransform	1519
EMeshToZMapConverter.WorldToZMapTransform	1520
EMeshToZMapConverter.ZMapToWorldTransform	1520

4.118. EMovingAverage Class	1521
EMovingAverage.Average	1522
EMovingAverage.EMovingAverage	1522
EMovingAverage.GetSize	1523
EMovingAverage.Reset	1524
EMovingAverage.SetSize	1525
EMovingAverage.SrcImage	1525
4.119. EObject Class	1526
EObject.GetHole	1527
EObject.HoleCount	1527
4.120. EObjectBasedCalibrationGenerator Class	1528
EObjectBasedCalibrationGenerator.CalibrationObjectScaleX	1531
EObjectBasedCalibrationGenerator.CalibrationObjectScaleY	1531
EObjectBasedCalibrationGenerator.CalibrationObjectScaleZ	1531
EObjectBasedCalibrationGenerator.CalibrationObjectSizeA	1532
EObjectBasedCalibrationGenerator.CalibrationObjectSizeB	1532
EObjectBasedCalibrationGenerator.CalibrationObjectSizeC	1533
EObjectBasedCalibrationGenerator.Compute	1533
EObjectBasedCalibrationGenerator.EObjectBasedCalibrationGenerator	1534
EObjectBasedCalibrationGenerator.GetCalibrationObjectType	1534
EObjectBasedCalibrationGenerator.Load	1535
EObjectBasedCalibrationGenerator.NumCalibrationPasses	1535
EObjectBasedCalibrationGenerator.operator=	1536
EObjectBasedCalibrationGenerator.PrecisionVsSpeedTradeOff	1536
EObjectBasedCalibrationGenerator.RangeX	1537
EObjectBasedCalibrationGenerator.RangeY	1537
EObjectBasedCalibrationGenerator.RangeZ	1537
EObjectBasedCalibrationGenerator.Save	1538
EObjectBasedCalibrationGenerator.SetCalibrationObjectScale	1539
EObjectBasedCalibrationGenerator.SetCalibrationObjectType	1540
4.121. EObjectBasedCalibrationModel Class	1541
EObjectBasedCalibrationModel.CalibrationError	1542
EObjectBasedCalibrationModel.CalibrationRelativeError	1542
EObjectBasedCalibrationModel.EObjectBasedCalibrationModel	1543
EObjectBasedCalibrationModel.IsInitialized	1543
EObjectBasedCalibrationModel.Load	1544
EObjectBasedCalibrationModel.operator=	1544
EObjectBasedCalibrationModel.Save	1545
EObjectBasedCalibrationModel.Type	1545
4.122. EObjectRunsIterator Class	1545
EObjectRunsIterator.EndX	1547
EObjectRunsIterator.EObjectRunsIterator	1547
EObjectRunsIterator.First	1548
EObjectRunsIterator.IsDone	1548
EObjectRunsIterator.Length	1549
EObjectRunsIterator.Next	1549
EObjectRunsIterator.operator=	1550
EObjectRunsIterator.StartX	1550
EObjectRunsIterator.Y	1551
4.123. EObjectSelection Class	1551
EObjectSelection.Add	1556
EObjectSelection.AddHole	1557
EObjectSelection.AddHoles	1558
EObjectSelection.AddHolesOfSelectedObjects	1558

EObjectSelection.AddLayer	1559
EObjectSelection.AddObject	1560
EObjectSelection.AddObjects	1560
EObjectSelection.AddObjectsUsingFloatFeature	1561
EObjectSelection.AddObjectsUsingIntegerFeature	1562
EObjectSelection.AddObjectsUsingRectangle	1564
EObjectSelection.AddObjectsUsingUnsignedIntegerFeature	1565
EObjectSelection.AddObjectUsingPosition	1566
EObjectSelection.AttachedImage	1567
EObjectSelection.Clear	1568
EObjectSelection.ClearFeatureCache	1568
EObjectSelection.ElementCount	1568
EObjectSelection.EObjectSelection	1569
EObjectSelection.FeatureAverage	1569
EObjectSelection.FeatureDeviation	1570
EObjectSelection.FeatureVariance	1570
EObjectSelection.FeretAngle	1571
EObjectSelection.FloatFeatureMaximum	1571
EObjectSelection.FloatFeatureMinimum	1572
EObjectSelection.GetElement	1573
EObjectSelection.GetFloatFeature	1573
EObjectSelection.GetIndexOfElement	1574
EObjectSelection.GetIntegerFeature	1574
EObjectSelection.GetUnsignedIntegerFeature	1575
EObjectSelection.IntegerFeatureMaximum	1576
EObjectSelection.IntegerFeatureMinimum	1576
EObjectSelection.IsSelected	1577
EObjectSelection.Remove	1578
EObjectSelection.RemoveHole	1578
EObjectSelection.RemoveHoles	1579
EObjectSelection.RemoveLayer	1580
EObjectSelection.RemoveObject	1581
EObjectSelection.RemoveObjectsUsingRectangle	1582
EObjectSelection.RemoveObjectUsingPosition	1583
EObjectSelection.RemoveSelectedHoles	1584
EObjectSelection.RemoveUsingFloatFeature	1584
EObjectSelection.RemoveUsingIntegerFeature	1585
EObjectSelection.RemoveUsingUnsignedIntegerFeature	1586
EObjectSelection.RenderMask	1587
EObjectSelection.Sort	1588
EObjectSelection.UnsignedIntegerFeatureMaximum	1588
EObjectSelection.UnsignedIntegerFeatureMinimum	1589
4.124. EOCR Class	1589
EOCR.AddChar	1597
EOCR.AddPatternFromImage	1597
EOCR.BuildObjects	1598
EOCR.CharGetDstX	1599
EOCR.CharGetDstY	1599
EOCR.CharGetHeight	1600
EOCR.CharGetOrgX	1600
EOCR.CharGetOrgY	1601
EOCR.CharGetTotalDstX	1601
EOCR.CharGetTotalDstY	1602
EOCR.CharGetTotalOrgX	1603
EOCR.CharGetTotalOrgY	1603

EOCR.CharGetWidth	1604
EOCR.CharSpacing	1604
EOCR.CompareAspectRatio	1605
EOCR.CutLargeChars	1605
EOCR.DrawChar	1606
EOCR.DrawChars	1607
EOCR.DrawCharsWithCurrentPen	1608
EOCR.DrawCharWithCurrentPen	1609
EOCR.EmptyChars	1610
EOCR.EOCR	1611
EOCR.FindAllChars	1612
EOCR.GetConfidenceRatio	1612
EOCR.GetFirstCharCode	1613
EOCR.GetFirstCharDistance	1613
EOCR.GetPatternBitmap	1614
EOCR.GetPatternClass	1614
EOCR.GetPatternCode	1615
EOCR.GetSecondCharCode	1615
EOCR.GetSecondCharDistance	1616
EOCR.HitChar	1616
EOCR.HitChars	1617
EOCR.LearnPattern	1618
EOCR.LearnPatterns	1619
EOCR.Load	1620
EOCR.MatchChar	1621
EOCR.MatchingMode	1622
EOCR.MaxCharHeight	1622
EOCR.MaxCharWidth	1623
EOCR.MinCharHeight	1623
EOCR.MinCharWidth	1624
EOCR.NewFont	1624
EOCR.NoiseArea	1625
EOCR.NumChars	1625
EOCR.NumPatterns	1626
EOCR.operator=	1626
EOCR.PatternHeight	1626
EOCR.PatternWidth	1627
EOCR.ReadText	1627
EOCR.ReadTextWide	1628
EOCR.Recognize	1629
EOCR.RecognizeWide	1630
EOCR.RelativeSpacing	1631
EOCR.RelativeThreshold	1632
EOCR.RemoveBorder	1632
EOCR.RemoveNarrowOrFlat	1633
EOCR.RemovePattern	1633
EOCR.Save	1634
EOCR.SegmentationMode	1634
EOCR.SetPatternClass	1635
EOCR.SetPatternCode	1635
EOCR.ShiftingMode	1636
EOCR.ShiftXTolerance	1636
EOCR.ShiftYtolerance	1637
EOCR.TextColor	1637
EOCR.Threshold	1638

EOCR.TrueThreshold	1638
4.125. EOCR2 Class	1639
EOCR2.AddCharactersToDatabase	1645
EOCR2.CharacterDatabase	1646
EOCR2.CharsHeight	1646
EOCR2.CharsMaxFragmentation	1647
EOCR2.CharsSpacingBias	1647
EOCR2.CharsWidthBias	1648
EOCR2.CharsWidthRange	1648
EOCR2.ClearCharacterDatabase	1649
EOCR2.ClearResult	1649
EOCR2.Detect	1650
EOCR2.DetectionDelta	1650
EOCR2.DetectionMethod	1651
EOCR2.DrawDetection	1651
EOCR2.DrawDetectionWithCurrentPen	1653
EOCR2.DrawRecognition	1654
EOCR2.DrawRecognitionWithCurrentPen	1656
EOCR2.DrawSegmentation	1657
EOCR2.DrawSegmentationWithCurrentPen	1658
EOCR2.EOCR2	1659
EOCR2.HitTestChar	1659
EOCR2.HitTestLine	1661
EOCR2.HitTestText	1662
EOCR2.HitTestWord	1663
EOCR2.Learn	1664
EOCR2.Load	1665
EOCR2.MaxVariation	1665
EOCR2.NumDetectionPasses	1666
EOCR2.operator=	1666
EOCR2.Read	1667
EOCR2.ReadText	1668
EOCR2.Recognize	1668
EOCR2.RelativeSpacesWidthRange	1669
EOCR2.Save	1669
EOCR2.SaveCharacterDatabase	1670
EOCR2.SegmentationMethod	1670
EOCR2.TextAngleRange	1671
EOCR2.TextPolarity	1671
EOCR2.TimeOut	1672
EOCR2.Topology	1673
4.126. EOCR2Char Class	1673
EOCR2Char.Bitmap	1674
EOCR2Char.BoundingBox	1675
EOCR2Char.Candidates	1675
EOCR2Char.EOCR2Char	1676
EOCR2Char.operator=	1676
EOCR2Char.Text	1677
EOCR2Char.TextCode	1677
4.127. EOCR2CharacterCluster Class	1678
EOCR2CharacterCluster.AddCharacter	1679
EOCR2CharacterCluster.CharacterCount	1679
EOCR2CharacterCluster.Characters	1680
EOCR2CharacterCluster.Clear	1680
EOCR2CharacterCluster.Code	1681

EOCR2CharacterCluster.EOCR2CharacterCluster	1681
EOCR2CharacterCluster.GetCharacter	1682
EOCR2CharacterCluster.operator=	1682
EOCR2CharacterCluster.RemoveCharacter	1683
4.128. EOCR2CharacterDatabase Class	1683
EOCR2CharacterDatabase.AddCharacter	1685
EOCR2CharacterDatabase.AddCharacters	1685
EOCR2CharacterDatabase.AddCluster	1686
EOCR2CharacterDatabase.AddClusters	1687
EOCR2CharacterDatabase.Characters	1687
EOCR2CharacterDatabase.ClearDatabase	1688
EOCR2CharacterDatabase.ClusterDatabase	1688
EOCR2CharacterDatabase.EOCR2CharacterDatabase	1689
EOCR2CharacterDatabase.GetCharacter	1689
EOCR2CharacterDatabase.operator=	1690
EOCR2CharacterDatabase.RemoveCharacter	1690
EOCR2CharacterDatabase.Save	1691
4.129. EOCR2DatabaseCharacter Class	1691
EOCR2DatabaseCharacter.Bitmap	1692
EOCR2DatabaseCharacter.CharacterCode	1692
EOCR2DatabaseCharacter.EOCR2DatabaseCharacter	1693
EOCR2DatabaseCharacter.operator=	1693
4.130. EOCR2Line Class	1694
EOCR2Line.BoundingBox	1695
EOCR2Line.EOCR2Line	1695
EOCR2Line.operator=	1696
EOCR2Line.Text	1696
EOCR2Line.Words	1697
4.131. EOCR2Text Class	1697
EOCR2Text.BoundingBox	1698
EOCR2Text.EOCR2Text	1698
EOCR2Text.Lines	1699
EOCR2Text.operator=	1699
EOCR2Text.Text	1700
4.132. EOCR2Word Class	1700
EOCR2Word.BoundingBox	1701
EOCR2Word.Characters	1702
EOCR2Word.EOCR2Word	1702
EOCR2Word.operator=	1703
EOCR2Word.Text	1703
4.133. EOCV Class	1703
EOCV.AccurateTextsLocationScores	1712
EOCV.AdjustCharsLocationRanges	1713
EOCV.AdjustCharsQualityRanges	1713
EOCV.AdjustShiftTolerances	1714
EOCV.AdjustTextsLocationRanges	1715
EOCV.AdjustTextsQualityRanges	1716
EOCV.ClearStatistics	1716
EOCV.ComputeDefaultTolerances	1717
EOCV.ContrastAverage	1717
EOCV.ContrastDeviation	1718
EOCV.ContrastTolerance	1718
EOCV.CreateTemplateChars	1719
EOCV.CreateTemplateObjects	1720

EOCV.CreateTemplateTexts	1720
EOCV.DeleteTemplateChars	1721
EOCV.DeleteTemplateObjects	1722
EOCV.DeleteTemplateTexts	1722
EOCV.Diagnostics	1723
EOCV.DrawTemplateChars	1723
EOCV.DrawTemplateCharsWithCurrentPen	1725
EOCV.DrawTemplateObjects	1726
EOCV.DrawTemplateObjectsWithCurrentPen	1727
EOCV.DrawTemplateTexts	1728
EOCV.DrawTemplateTextsChars	1730
EOCV.DrawTemplateTextsCharsWithCurrentPen	1731
EOCV.DrawTemplateTextsWithCurrentPen	1732
EOCV.DrawText	1733
EOCV.DrawTextChars	1735
EOCV.DrawTextCharsWithCurrentPen	1737
EOCV.DrawTexts	1738
EOCV.DrawTextsChars	1739
EOCV.DrawTextsCharsWithCurrentPen	1741
EOCV.DrawTextsWithCurrentPen	1742
EOCV.DrawTextWithCurrentPen	1743
EOCV.EOCV	1744
EOCV.FreeCharCount	1745
EOCV.GatherTextsCharsParameters	1745
EOCV.GatherTextsParameters	1746
EOCV.GetFreeChar	1746
EOCV.GetNumTextChars	1747
EOCV.GetText	1748
EOCV.GetTextCharParameters	1748
EOCV.GetTextCharPoint	1749
EOCV.GetTextParameters	1750
EOCV.GetTextPoint	1751
EOCV.Inspect	1753
EOCV.Learn	1754
EOCV.Load	1755
EOCV.LocationMode	1755
EOCV.NormalizeLocationScore	1756
EOCV.NumTexts	1756
EOCV.ReduceLocationScore	1757
EOCV.ResampleChars	1757
EOCV.SampleBackground	1758
EOCV.SampleContrast	1758
EOCV.SampleForeground	1759
EOCV.SampleThreshold	1759
EOCV.Save	1760
EOCV.ScatterTextsCharsParameters	1760
EOCV.ScatterTextsParameters	1761
EOCV.SelectSampleTexts	1762
EOCV.SelectSampleTextsChars	1763
EOCV.SelectTemplateChars	1764
EOCV.SelectTemplateObjects	1765
EOCV.SelectTemplateTexts	1765
EOCV.SetFreeChar	1766
EOCV.SetText	1767
EOCV.SetTextCharParameters	1768

EOCV.SetTextParameters	1768
EOCV.StatisticsCount	1769
EOCV.TemplateBackground	1770
EOCV.TemplateContrast	1770
EOCV.TemplateForeground	1771
EOCV.TemplateImage	1771
EOCV.TemplateThreshold	1772
EOCV.UpdateStatistics	1772
EOCV.UsedQualityIndicators	1773
EOCV.WhiteOnBlack	1773
4.134. EOCVChar Class	1774
EOCVChar.BackgroundAreaAverage	1780
EOCVChar.BackgroundAreaDeviation	1780
EOCVChar.BackgroundAreaTolerance	1780
EOCVChar.BackgroundSumAverage	1781
EOCVChar.BackgroundSumDeviation	1781
EOCVChar.BackgroundSumTolerance	1782
EOCVChar.Correlation	1782
EOCVChar.CorrelationAverage	1782
EOCVChar.CorrelationDeviation	1783
EOCVChar.CorrelationTolerance	1783
EOCVChar.Diagnostics	1784
EOCVChar.EOCVChar	1784
EOCVChar.ForegroundAreaAverage	1785
EOCVChar.ForegroundAreaDeviation	1785
EOCVChar.ForegroundAreaTolerance	1785
EOCVChar.ForegroundSumAverage	1786
EOCVChar.ForegroundSumDeviation	1786
EOCVChar.ForegroundSumTolerance	1787
EOCVChar.LocationScoreAverage	1787
EOCVChar.LocationScoreDeviation	1787
EOCVChar.LocationScoreTolerance	1788
EOCVChar.MarginWidth	1788
EOCVChar.NumContourPoints	1789
EOCVChar.operator=	1789
EOCVChar.ResetParameters	1790
EOCVChar.SampleBackgroundArea	1790
EOCVChar.SampleBackgroundSum	1791
EOCVChar.SampleForegroundArea	1791
EOCVChar.SampleForegroundSum	1791
EOCVChar.SampleLocationScore	1792
EOCVChar.Selected	1792
EOCVChar.ShiftX	1793
EOCVChar.ShiftXAverage	1793
EOCVChar.ShiftXBias	1793
EOCVChar.ShiftXDeviation	1794
EOCVChar.ShiftXMax	1794
EOCVChar.ShiftXMin	1795
EOCVChar.ShiftXStride	1795
EOCVChar.ShiftXTolerance	1795
EOCVChar.ShiftY	1796
EOCVChar.ShiftYAverage	1796
EOCVChar.ShiftYBias	1797
EOCVChar.ShiftYDeviation	1797
EOCVChar.ShiftYMax	1797

EOCVChar.ShiftYMin	1798
EOCVChar.ShiftYStride	1798
EOCVChar.ShiftYTolerance	1799
EOCVChar.TemplateBackgroundArea	1799
EOCVChar.TemplateBackgroundSum	1799
EOCVChar.TemplateForegroundArea	1800
EOCVChar.TemplateForegroundSum	1800
EOCVChar.TemplateLocationScore	1801
EOCVChar.WhiteOnBlack	1801
4.135. EOCVText Class	1802
EOCVText.BackgroundAreaAverage	1810
EOCVText.BackgroundAreaDeviation	1811
EOCVText.BackgroundAreaTolerance	1811
EOCVText.BackgroundSumAverage	1811
EOCVText.BackgroundSumDeviation	1812
EOCVText.BackgroundSumTolerance	1812
EOCVText.Correlation	1813
EOCVText.CorrelationAverage	1813
EOCVText.CorrelationDeviation	1813
EOCVText.CorrelationTolerance	1814
EOCVText.Diagnostics	1814
EOCVText.EOCVText	1815
EOCVText.ForegroundAreaAverage	1815
EOCVText.ForegroundAreaDeviation	1816
EOCVText.ForegroundAreaTolerance	1816
EOCVText.ForegroundSumAverage	1816
EOCVText.ForegroundSumDeviation	1817
EOCVText.ForegroundSumTolerance	1817
EOCVText.IsotropicScaling	1818
EOCVText.LocationScoreAverage	1818
EOCVText.LocationScoreDeviation	1818
EOCVText.LocationScoreTolerance	1819
EOCVText.MarginWidth	1819
EOCVText.NumContourPoints	1820
EOCVText.operator=	1820
EOCVText.ResetParameters	1821
EOCVText.SampleBackgroundArea	1821
EOCVText.SampleBackgroundSum	1821
EOCVText.SampleForegroundArea	1822
EOCVText.SampleForegroundSum	1822
EOCVText.SampleLocationScore	1823
EOCVText.ScaleX	1823
EOCVText.ScaleXAverage	1823
EOCVText.ScaleXBias	1824
EOCVText.ScaleXCount	1824
EOCVText.ScaleXDeviation	1825
EOCVText.ScaleXMax	1825
EOCVText.ScaleXMin	1825
EOCVText.ScaleXTolerance	1826
EOCVText.ScaleY	1826
EOCVText.ScaleYAverage	1827
EOCVText.ScaleYBias	1827
EOCVText.ScaleYCount	1827
EOCVText.ScaleYDeviation	1828
EOCVText.ScaleYMax	1828

EOCVText.ScaleYMin	1829
EOCVText.ScaleYTolerance	1829
EOCVText.Selected	1829
EOCVText.Shear	1830
EOCVText.ShearAverage	1830
EOCVText.ShearBias	1831
EOCVText.ShearCount	1831
EOCVText.ShearDeviation	1831
EOCVText.ShearMax	1832
EOCVText.ShearMin	1832
EOCVText.ShearTolerance	1833
EOCVText.ShiftX	1833
EOCVText.ShiftXAverage	1833
EOCVText.ShiftXBias	1834
EOCVText.ShiftXDeviation	1834
EOCVText.ShiftXMax	1835
EOCVText.ShiftXMin	1835
EOCVText.ShiftXStride	1835
EOCVText.ShiftXTolerance	1836
EOCVText.ShiftY	1836
EOCVText.ShiftYAverage	1837
EOCVText.ShiftYBias	1837
EOCVText.ShiftYDeviation	1837
EOCVText.ShiftYMax	1838
EOCVText.ShiftYMin	1838
EOCVText.ShiftYStride	1839
EOCVText.ShiftYTolerance	1839
EOCVText.Skew	1839
EOCVText.SkewAverage	1840
EOCVText.SkewBias	1840
EOCVText.SkewCount	1841
EOCVText.SkewDeviation	1841
EOCVText.SkewMax	1841
EOCVText.SkewMin	1842
EOCVText.SkewTolerance	1842
EOCVText.TemplateBackgroundArea	1843
EOCVText.TemplateBackgroundSum	1843
EOCVText.TemplateForegroundArea	1843
EOCVText.TemplateForegroundSum	1844
EOCVText.TemplateLocationScore	1844
4.136. EPathVector Class	1845
EPathVector.AddElement	1846
EPathVector.Closed	1847
EPathVector.Draw	1847
EPathVector.DrawWithCurrentPen	1849
EPathVector.EPathVector	1850
EPathVector.GetElement	1850
EPathVector.operator[]	1851
EPathVector.operator=	1851
EPathVector.RawDataPtr	1852
EPathVector.SetElement	1852
4.137. EPatternFinder Class	1853
EPatternFinder.AngleBias	1857
EPatternFinder.AngleSearchExtent	1857
EPatternFinder.AngleTolerance	1857

EPatternFinder.AutoTransitionThickness	1858
EPatternFinder.ContrastMode	1858
EPatternFinder.CopyLearntPattern	1859
EPatternFinder.DrawModel	1859
EPatternFinder.DrawModelWithCurrentPen	1861
EPatternFinder.EPatternFinder	1861
EPatternFinder.Find	1862
EPatternFinder.FindExtension	1863
EPatternFinder.ForcedThreshold	1863
EPatternFinder.Interpolate	1864
EPatternFinder.Learn	1864
EPatternFinder.LearningDone	1865
EPatternFinder.LightBalance	1866
EPatternFinder.LocalSearchMode	1866
EPatternFinder.MaxFeaturePoints	1867
EPatternFinder.MaxInstances	1867
EPatternFinder.MinFeaturePoints	1868
EPatternFinder.MinScore	1868
EPatternFinder.operator=	1869
EPatternFinder.PatternType	1869
EPatternFinder.Pivot	1870
EPatternFinder.ReductionMode	1870
EPatternFinder.ReductionStrength	1871
EPatternFinder.ScaleBias	1872
EPatternFinder.ScaleSearchExtent	1872
EPatternFinder.ScaleTolerance	1872
EPatternFinder.ThinStructureMode	1873
EPatternFinder.TransitionThickness	1873
EPatternFinder.Type	1874
EPatternFinder.XSearchExtent	1874
EPatternFinder.YSearchExtent	1875
4.138. EPeakVector Class	1875
EPeakVector.AddElement	1876
EPeakVector.EPeakVector	1877
EPeakVector.GetElement	1877
EPeakVector.operator[]	1878
EPeakVector.operator=	1878
EPeakVector.RawDataPtr	1879
EPeakVector.SetElement	1879
4.139. EPlaneCropper Class	1880
EPlaneCropper.Crop	1881
EPlaneCropper.EPlaneCropper	1882
EPlaneCropper.Load	1883
EPlaneCropper.operator=	1883
EPlaneCropper.Plane	1884
EPlaneCropper.Save	1884
4.140. EPlaneFinder Class	1885
EPlaneFinder.DisableDecimator	1887
EPlaneFinder.EnableDecimator	1888
EPlaneFinder.EPlaneFinder	1888
EPlaneFinder.ExpectedCloudInliersRatio	1889
EPlaneFinder.Find	1889
EPlaneFinder.GetNormal	1890
EPlaneFinder.IsDecimatorEnabled	1890
EPlaneFinder.IsNormalSet	1891

EPlaneFinder.Load	1891
EPlaneFinder.MaxDeviation	1892
EPlaneFinder.NormalTolerance	1892
EPlaneFinder.operator=	1892
EPlaneFinder.Save	1893
EPlaneFinder.SetNormal	1893
EPlaneFinder.UnsetNormal	1894
4.141. EPlaneFitter Class	1895
EPlaneFitter.EPlaneFitter	1896
EPlaneFitter.Fit	1896
EPlaneFitter.Load	1897
EPlaneFitter.MinSampleCount	1898
EPlaneFitter.operator=	1898
EPlaneFitter.Save	1899
4.142. EPoint Class	1899
EPoint.Area	1902
EPoint.Argument	1902
EPoint.Center	1903
EPoint.CopyTo	1903
EPoint.Distance	1904
EPoint.Dot	1905
EPoint.EPoint	1905
EPoint.MidPoint	1906
EPoint.Modulus	1907
EPoint.operator-	1907
EPoint.operator!=	1908
EPoint.operator*	1908
EPoint.operator/	1909
EPoint.operator+	1909
EPoint.operator=	1910
EPoint.operator==	1910
EPoint.Project	1911
EPoint.Rotate	1911
EPoint.SetCenterXY	1912
EPoint.Square	1912
EPoint.SquaredDistance	1913
EPoint.X	1913
EPoint.Y	1914
4.143. EPointCloud Class	1914
EPointCloud.AddPoint	1916
EPointCloud.AddPointCloud	1916
EPointCloud.AddPoints	1917
EPointCloud.Clear	1917
EPointCloud.EPointCloud	1918
EPointCloud.FillPointsBuffer	1918
EPointCloud.GetPoint	1919
EPointCloud.Load	1920
EPointCloud.LoadPCD	1920
EPointCloud.NumPoints	1921
EPointCloud.operator=	1921
EPointCloud.PointsBuffer	1922
EPointCloud.Save	1922
EPointCloud.SavePCD	1923
EPointCloud.Serialize	1923
4.144. EPointCloudFactory Class	1924

EPointCloudFactory.CreateCubicPointCloud	1924
EPointCloudFactory.CreateRectangularPointCloud	1925
EPointCloudFactory.CreateSphericPointCloud	1926
4.145. EPointCloudStatistics Class	1927
EPointCloudStatistics.GetPointCloudBounds	1928
EPointCloudStatistics.GetPointCloudCentroid	1929
4.146. EPointCloudToZMapConverter Class	1930
EPointCloudToZMapConverter.Convert	1935
EPointCloudToZMapConverter.EnableFillMode	1936
EPointCloudToZMapConverter.EPointCloudToZMapConverter	1937
EPointCloudToZMapConverter.Extension	1937
EPointCloudToZMapConverter.FillUndefinedPixelsDirection	1938
EPointCloudToZMapConverter.FillUndefinedPixelsMethod	1938
EPointCloudToZMapConverter.IsFillModeEnabled	1938
EPointCloudToZMapConverter.Load	1939
EPointCloudToZMapConverter.MapHeight	1939
EPointCloudToZMapConverter.MapWidth	1940
EPointCloudToZMapConverter.MapXResolution	1940
EPointCloudToZMapConverter.MapYResolution	1941
EPointCloudToZMapConverter.MapZResolution	1941
EPointCloudToZMapConverter.operator=	1941
EPointCloudToZMapConverter.operator==	1942
EPointCloudToZMapConverter.OrientationVector	1942
EPointCloudToZMapConverter.OrientationVectorMode	1943
EPointCloudToZMapConverter.Origin	1943
EPointCloudToZMapConverter.ReferencePlane	1944
EPointCloudToZMapConverter.ReferencePlaneMode	1944
EPointCloudToZMapConverter.Save	1945
EPointCloudToZMapConverter.SetFillMode	1945
EPointCloudToZMapConverter.SetMapSize	1946
EPointCloudToZMapConverter.SetMapXYResolution	1947
EPointCloudToZMapConverter.UnsetMapSize	1947
EPointCloudToZMapConverter.UnsetMapXYResolution	1948
EPointCloudToZMapConverter.UnsetMapZResolution	1948
EPointCloudToZMapConverter.UnsetOrigin	1949
EPointCloudToZMapConverter.UnsetWorldToZMapTransform	1949
EPointCloudToZMapConverter.WorldToZMapTransform	1950
EPointCloudToZMapConverter.ZMapToWorldTransform	1950
4.147. EPointGauge Class	1951
EPointGauge.Active	1954
EPointGauge.Center	1955
EPointGauge.CopyTo	1955
EPointGauge.Drag	1956
EPointGauge.Draw	1956
EPointGauge.DrawWithCurrentPen	1957
EPointGauge.EPointGauge	1958
EPointGauge.GetMeasuredPeak	1959
EPointGauge.GetMeasuredPoint	1959
EPointGauge.HitTest	1960
EPointGauge.HVConstraint	1961
EPointGauge.Measure	1961
EPointGauge.MinAmplitude	1962
EPointGauge.MinArea	1962
EPointGauge.NumMeasuredPoints	1963
EPointGauge.operator=	1963

EPointGauge.Plot	1964
EPointGauge.PlotWithCurrentPen	1965
EPointGauge.Process	1966
EPointGauge.RectangularSamplingArea	1966
EPointGauge.SetCenterXY	1967
EPointGauge.SetTolerances	1968
EPointGauge.Smoothing	1968
EPointGauge.Thickness	1969
EPointGauge.Threshold	1969
EPointGauge.Tolerance	1970
EPointGauge.ToleranceAngle	1970
EPointGauge.TransitionChoice	1971
EPointGauge.TransitionIndex	1972
EPointGauge.TransitionType	1972
EPointGauge.Type	1973
EPointGauge.Valid	1973
4.148. EPointShape Class	1973
EPointShape.Center	1975
EPointShape.CenterX	1976
EPointShape.CenterY	1976
EPointShape.Closest	1977
EPointShape.CopyTo	1977
EPointShape.Drag	1978
EPointShape.Draw	1978
EPointShape.DrawWithCurrentPen	1979
EPointShape.HitTest	1980
EPointShape.operator!=	1980
EPointShape.operator=	1981
EPointShape.operator==	1981
EPointShape.SetCenterXY	1982
EPointShape.Type	1982
4.149. EPolygonRegion Class	1983
EPolygonRegion.Drag	1984
EPolygonRegion.EPolygonRegion	1985
EPolygonRegion.HitTest	1986
EPolygonRegion.InsertPoint	1987
EPolygonRegion.Load	1988
EPolygonRegion.operator!=	1988
EPolygonRegion.operator=	1989
EPolygonRegion.operator==	1989
EPolygonRegion.Points	1990
EPolygonRegion.RemovePoint	1990
EPolygonRegion.Rotate	1991
EPolygonRegion.Save	1992
EPolygonRegion.Scale	1992
EPolygonRegion.Translate	1993
4.150. EPrincipalAxisExtractor Class	1994
EPrincipalAxisExtractor.EPrincipalAxisExtractor	1995
EPrincipalAxisExtractor.Extract	1996
EPrincipalAxisExtractor.HasReferenceTransformSet	1996
EPrincipalAxisExtractor.Load	1997
EPrincipalAxisExtractor.operator=	1997
EPrincipalAxisExtractor.ReferenceTransform	1998
EPrincipalAxisExtractor.Save	1998
EPrincipalAxisExtractor.UnsetReferenceTransform	1999

4.151. EPseudoColorLookup Class	1999
EPseudoColorLookup.EPseudoColorLookup	2000
EPseudoColorLookup.SetShading	2000
4.152. EQRCode Class	2001
EQRCode.DecodedStream	2003
EQRCode.Draw	2003
EQRCode.DrawWithCurrentPen	2004
EQRCode.EQRCode	2005
EQRCode.Geometry	2005
EQRCode.IsDecodingReliable	2006
EQRCode.Level	2006
EQRCode.Model	2006
EQRCode.operator=	2007
EQRCode.UnusedErrorCorrection	2007
EQRCode.Version	2008
4.153. EQRCodeDecodedStream Class	2008
EQRCodeDecodedStream.ApplicationIndicator	2009
EQRCodeDecodedStream.CodingMode	2010
EQRCodeDecodedStream.DecodedStreamParts	2010
EQRCodeDecodedStream.EQRCodeDecodedStream	2010
EQRCodeDecodedStream.operator=	2011
EQRCodeDecodedStream.RawBitstream	2011
4.154. EQRCodeDecodedStreamPart Class	2012
EQRCodeDecodedStreamPart.DecodedData	2013
EQRCodeDecodedStreamPart.Encoding	2013
EQRCodeDecodedStreamPart.EQRCodeDecodedStreamPart	2013
EQRCodeDecodedStreamPart.operator=	2014
4.155. EQRCodeGeometry Class	2014
EQRCodeGeometry.Draw	2015
EQRCodeGeometry.DrawWithCurrentPen	2016
EQRCodeGeometry.EQRCodeGeometry	2017
EQRCodeGeometry.FinderPatternCenters	2018
EQRCodeGeometry.operator=	2018
EQRCodeGeometry.Position	2019
4.156. EQRCodeReader Class	2019
EQRCodeReader.CellPolarityConfidenceThreshold	2022
EQRCodeReader.Decode	2022
EQRCodeReader.Detect	2023
EQRCodeReader.DetectionMethod	2023
EQRCodeReader.DetectionTradeOff	2024
EQRCodeReader.EQRCodeReader	2024
EQRCodeReader.FilterOutUnreliablyDecodedQRCodes	2025
EQRCodeReader.ForegroundDetectionThreshold	2025
EQRCodeReader.MaximumVersion	2026
EQRCodeReader.MinimumIsotropy	2026
EQRCodeReader.MinimumScore	2027
EQRCodeReader.MinimumVersion	2027
EQRCodeReader.PerspectiveMode	2028
EQRCodeReader.Read	2028
EQRCodeReader.ScanPrecision	2028
EQRCodeReader.SearchedModels	2029
EQRCodeReader.SearchField	2029
EQRCodeReader.TimeOut	2030
4.157. EQuadrangle Class	2030

EQuadrangle.Draw	2032
EQuadrangle.DrawWithCurrentPen	2033
EQuadrangle.EQuadrangle	2034
EQuadrangle.GetPoint	2035
EQuadrangle.GetSideAngle	2035
EQuadrangle.GravityCenter	2036
EQuadrangle.IsInside	2036
EQuadrangle.operator=	2037
EQuadrangle.OverLaps	2037
EQuadrangle.SetPoint	2038
4.158. EQuadrilateral Class	2038
EQuadrilateral.Corners	2039
EQuadrilateral.EQuadrilateral	2039
EQuadrilateral.operator=	2040
4.159. ERandomDecimator Class	2041
ERandomDecimator.Decimate	2042
ERandomDecimator.ERandomDecimator	2042
ERandomDecimator.NumberOfPoints	2043
ERandomDecimator.operator=	2043
ERandomDecimator.Serialize	2044
4.160. ERectangle Class	2044
ERectangle.CopyTo	2046
ERectangle.ERectangle	2047
ERectangle.GetCorners	2048
ERectangle.GetEdges	2049
ERectangle.GetMidEdges	2049
ERectangle.GetPoint	2050
ERectangle.operator=	2051
ERectangle.SetFromOppositeCorners	2051
ERectangle.SetFromOriginMiddleEnd	2052
ERectangle.SetFromThreeCorners	2052
ERectangle.SetFromTwoPoints	2053
ERectangle.SetSize	2054
ERectangle.SizeX	2054
ERectangle.SizeY	2055
4.161. ERectangleGauge Class	2055
ERectangleGauge.Active	2061
ERectangleGauge.ActiveEdges	2062
ERectangleGauge.AddSkipRange	2062
ERectangleGauge.AverageDistance	2063
ERectangleGauge.CopyTo	2063
ERectangleGauge.DisableInnerFiltering	2064
ERectangleGauge.Drag	2065
ERectangleGauge.Draw	2065
ERectangleGauge.DrawWithCurrentPen	2066
ERectangleGauge.ERectangleGauge	2067
ERectangleGauge.FilteringThreshold	2068
ERectangleGauge.GetMeasuredPoint	2068
ERectangleGauge.GetMinNumFitSamples	2069
ERectangleGauge.GetSampleX	2070
ERectangleGauge.GetSampleX	2071
ERectangleGauge.GetSampleY	2072
ERectangleGauge.GetSampleY	2073
ERectangleGauge.GetSkipRange	2074
ERectangleGauge.HitTest	2074

ERectangleGauge.HVConstraint	2075
ERectangleGauge.InnerFilteringEnabled	2075
ERectangleGauge.InnerFilteringThreshold	2076
ERectangleGauge.KnownAngle	2076
ERectangleGauge.Measure	2077
ERectangleGauge.MeasuredRectangle	2078
ERectangleGauge.MeasureSample	2078
ERectangleGauge.MeasureWithoutFitting	2079
ERectangleGauge.MinAmplitude	2079
ERectangleGauge.MinArea	2080
ERectangleGauge.NumFilteringPasses	2080
ERectangleGauge.NumSamples	2081
ERectangleGauge.NumSamplesX	2081
ERectangleGauge.NumSamplesX	2082
ERectangleGauge.NumSamplesY	2082
ERectangleGauge.NumSamplesY	2083
ERectangleGauge.NumSkipRanges	2083
ERectangleGauge.NumValidSamples	2083
ERectangleGauge.operator=	2084
ERectangleGauge.Plot	2084
ERectangleGauge.PlotWithCurrentPen	2086
ERectangleGauge.Process	2087
ERectangleGauge.RectangularSamplingArea	2087
ERectangleGauge.RemoveAllSkipRanges	2088
ERectangleGauge.RemoveSkipRange	2088
ERectangleGauge.SamplingStep	2089
ERectangleGauge.SetMinNumFitSamples	2089
ERectangleGauge.Smoothing	2090
ERectangleGauge.Thickness	2091
ERectangleGauge.Threshold	2091
ERectangleGauge.Tolerance	2092
ERectangleGauge.TransitionChoice	2092
ERectangleGauge.TransitionIndex	2093
ERectangleGauge.TransitionType	2093
ERectangleGauge.Type	2094
ERectangleGauge.Valid	2094
4.162. ERectangleRegion Class	2095
ERectangleRegion.Angle	2097
ERectangleRegion.Center	2097
ERectangleRegion.Drag	2098
ERectangleRegion.ERectangleRegion	2099
ERectangleRegion.Height	2100
ERectangleRegion.HitTest	2101
ERectangleRegion.Load	2102
ERectangleRegion.operator!=	2102
ERectangleRegion.operator=	2103
ERectangleRegion.operator==	2103
ERectangleRegion.Rotate	2104
ERectangleRegion.Save	2104
ERectangleRegion.Scale	2105
ERectangleRegion.Translate	2106
ERectangleRegion.Width	2106
4.163. ERectangleShape Class	2107
ERectangleShape.Angle	2110
ERectangleShape.Center	2110

ERectangleShape.CenterX	2110
ERectangleShape.CenterY	2111
ERectangleShape.Closest	2111
ERectangleShape.CopyTo	2112
ERectangleShape.Drag	2112
ERectangleShape.Draw	2113
ERectangleShape.DrawWithCurrentPen	2114
ERectangleShape.GetCorners	2114
ERectangleShape.GetEdges	2115
ERectangleShape.GetMidEdges	2116
ERectangleShape.GetPoint	2117
ERectangleShape.HitTest	2117
ERectangleShape.operator=	2118
ERectangleShape.Rectangle	2118
ERectangleShape.Scale	2119
ERectangleShape.SetCenterXY	2119
ERectangleShape.SetFromOppositeCorners	2120
ERectangleShape.SetFromOriginMiddleEnd	2120
ERectangleShape.SetFromThreeCorners	2121
ERectangleShape.SetFromTwoPoints	2122
ERectangleShape.SetSize	2122
ERectangleShape.SizeX	2123
ERectangleShape.SizeY	2123
ERectangleShape.Type	2124
4.164. ERectangularCropper Class	2124
ERectangularCropper.Crop	2125
ERectangularCropper.ERectangularCropper	2125
ERectangularCropper.operator=	2126
4.165. EReferencelImageSegmenter Class	2127
ERreferencelImageSegmenter.BlackLayerEncoded	2128
ERreferencelImageSegmenter.BlackLayerIndex	2129
ERreferencelImageSegmenter.ReferencelImageBW16	2129
ERreferencelImageSegmenter.ReferencelImageBW8	2129
ERreferencelImageSegmenter.ReferencelImageC24	2130
ERreferencelImageSegmenter.WhiteLayerEncoded	2130
ERreferencelImageSegmenter.WhiteLayerIndex	2131
4.166. ERegion Class	2131
ERegion.BoundingBoxHeight	2134
ERegion.BoundingBoxOrgX	2134
ERegion.BoundingBoxOrgY	2135
ERegion.BoundingBoxWidth	2135
ERegion.CropRuns	2136
ERegion.Drag	2136
ERegion.Draw	2137
ERegion.DrawContour	2139
ERegion.DrawContourWithCurrentPen	2140
ERegion.DrawHandles	2140
ERegion.DrawHandlesWithCurrentPen	2142
ERegion.DrawWithCurrentPen	2142
ERegion.EditionMode	2143
ERegion.ERegion	2144
ERegion.HitTest	2145
ERegion.Intersection	2146
ERegion.Load	2147
ERegion.operator!=	2147

ERegion.operator=	2148
ERegion.operator==	2148
ERegion.Prepare	2149
ERegion.Runs	2151
ERegion.Save	2151
ERegion.Subtraction	2152
ERegion.ToImage	2152
ERegion.Union	2153
4.167. EROIBW1 Class	2154
EROIBW1.EROIBW1	2155
EROIBW1.FirstSubROI	2156
EROIBW1.GetBitIndex	2156
EROIBW1.GetNextROI	2157
EROIBW1.GetPixel	2157
EROIBW1.NextSiblingROI	2158
EROIBW1.operator=	2158
EROIBW1.Parent	2159
EROIBW1.Serialize	2159
EROIBW1.SetPixel	2160
EROIBW1.TopParent	2160
4.168. EROIBW16 Class	2161
EROIBW16.EROIBW16	2162
EROIBW16.FirstSubROI	2163
EROIBW16.GetNextROI	2163
EROIBW16.GetPixel	2163
EROIBW16.NextSiblingROI	2164
EROIBW16.operator=	2165
EROIBW16.Parent	2165
EROIBW16.Serialize	2165
EROIBW16.SetPixel	2166
EROIBW16.TopParent	2167
4.169. EROIBW32 Class	2167
EROIBW32.EROIBW32	2168
EROIBW32.FirstSubROI	2169
EROIBW32.GetNextROI	2169
EROIBW32.GetPixel	2170
EROIBW32.NextSiblingROI	2171
EROIBW32.operator=	2171
EROIBW32.Parent	2171
EROIBW32.Serialize	2172
EROIBW32.SetPixel	2172
EROIBW32.TopParent	2173
4.170. EROIBW8 Class	2174
EROIBW8.EROIBW8	2175
EROIBW8.FirstSubROI	2175
EROIBW8.GetNextROI	2176
EROIBW8.GetPixel	2176
EROIBW8.NextSiblingROI	2177
EROIBW8.operator=	2178
EROIBW8.Parent	2178
EROIBW8.Serialize	2178
EROIBW8.SetPixel	2179
EROIBW8.TopParent	2180
4.171. EROIC15 Class	2180

EROIC15.EROIC15	2181
EROIC15.FirstSubROI	2182
EROIC15.GetNextROI	2182
EROIC15.GetPixel	2183
EROIC15.NextSiblingROI	2184
EROIC15.operator=	2184
EROIC15.Parent	2184
EROIC15.Serialize	2185
EROIC15.SetPixel	2185
EROIC15.TopParent	2186
4.172. EROIC16 Class	2187
EROIC16.EROIC16	2188
EROIC16.FirstSubROI	2188
EROIC16.GetNextROI	2189
EROIC16.GetPixel	2189
EROIC16.NextSiblingROI	2190
EROIC16.operator=	2191
EROIC16.Parent	2191
EROIC16.Serialize	2191
EROIC16.SetPixel	2192
EROIC16.TopParent	2193
4.173. EROIC24 Class	2193
EROIC24.EROIC24	2194
EROIC24.FirstSubROI	2195
EROIC24.GetNextROI	2195
EROIC24.GetPixel	2196
EROIC24.NextSiblingROI	2197
EROIC24.operator=	2197
EROIC24.Parent	2197
EROIC24.Serialize	2198
EROIC24.SetPixel	2198
EROIC24.TopParent	2199
4.174. EROIC24A Class	2200
EROIC24A.EROIC24A	2201
EROIC24A.FirstSubROI	2201
EROIC24A.GetNextROI	2202
EROIC24A.GetPixel	2202
EROIC24A.NextSiblingROI	2203
EROIC24A.operator=	2204
EROIC24A.Parent	2204
EROIC24A.Serialize	2204
EROIC24A.SetPixel	2205
EROIC24A.TopParent	2206
4.175. EROIC48 Class	2206
EROIC48.EROIC48	2207
EROIC48.FirstSubROI	2208
EROIC48.GetNextROI	2208
EROIC48.GetPixel	2209
EROIC48.NextSiblingROI	2210
EROIC48.operator=	2210
EROIC48.Parent	2210
EROIC48.Serialize	2211
EROIC48.SetPixel	2211
EROIC48.TopParent	2212

4.176. ERotatedBoundingBox Class	2213
ERotatedBoundingBox.Angle	2214
ERotatedBoundingBox.Center	2215
ERotatedBoundingBox.CenterX	2215
ERotatedBoundingBox.CenterY	2216
ERotatedBoundingBox.Draw	2216
ERotatedBoundingBox.DrawWithCurrentPen	2218
ERotatedBoundingBox.ERotatedBoundingBox	2219
ERotatedBoundingBox.Height	2220
ERotatedBoundingBox.LocalToGlobalBox	2220
ERotatedBoundingBox.LocalToGlobalPoint	2221
ERotatedBoundingBox.operator=	2221
ERotatedBoundingBox.Quadrangle	2222
ERotatedBoundingBox.Translate	2222
ERotatedBoundingBox.Width	2223
4.177. ESamplePoint Class	2223
ESamplePoint.ESamplePoint	2224
ESamplePoint.IsOutlier	2224
ESamplePoint.IsValid	2225
ESamplePoint.operator=	2225
ESamplePoint.Position	2226
4.178. EScaleCalibrationModel Class	2226
EScaleCalibrationModel.EScaleCalibrationModel	2227
EScaleCalibrationModel.FactorX	2228
EScaleCalibrationModel.FactorY	2229
EScaleCalibrationModel.FactorZ	2229
EScaleCalibrationModel.Load	2229
EScaleCalibrationModel.operator=	2230
EScaleCalibrationModel.operator==	2230
EScaleCalibrationModel.Save	2231
EScaleCalibrationModel.Type	2231
4.179. ESearchParamsType Class	2232
ESearchParamsType.AddContrast	2235
ESearchParamsType.AddFamily	2235
ESearchParamsType.AddFlipping	2236
ESearchParamsType.AddLogicalSize	2236
ESearchParamsType.ClearContrast	2237
ESearchParamsType.ClearFamily	2237
ESearchParamsType.ClearFlipping	2237
ESearchParamsType.ClearLogicalSize	2238
ESearchParamsType.ContrastCount	2238
ESearchParamsType.FamilyCount	2239
ESearchParamsType.FlippingCount	2239
ESearchParamsType.GetContrast	2239
ESearchParamsType.GetFamily	2240
ESearchParamsType.GetFlipping	2240
ESearchParamsType.GetLogicalSize	2241
ESearchParamsType.LogicalSizeCount	2241
ESearchParamsType.RemoveContrast	2242
ESearchParamsType.RemoveFamily	2242
ESearchParamsType.RemoveFlipping	2243
ESearchParamsType.RemoveLogicalSize	2243
4.180. ESerializer Class	2244
ESerializer.Close	2245

ESerializer.CreateFileReader	2245
ESerializer.CreateFileWriter	2246
ESerializer.CreateMemoryReader	2247
ESerializer.CreateMemoryWriter	2247
ESerializer.Writing	2248
4.181. EShape Class	2249
EShape.Active	2255
EShape.ActiveRecursive	2255
EShape.ActualShape	2255
EShape.ActualShapeRecursive	2256
EShape.Attach	2256
EShape.Closest	2257
EShape.ClosestShape	2257
EShape.Detach	2258
EShape.DetachDaughters	2258
EShape.DisableBehaviorFilter	2258
EShape.DisableTypeFilter	2259
EShape.Drag	2259
EShape.Dragable	2260
EShape.DragableRecursive	2260
EShape.Draw	2261
EShape.DrawWithCurrentPen	2262
EShape.EnableBehaviorFilter	2262
EShape.EnableTypeFilter	2263
EShape.GetAllocated	2264
EShape.GetDaughter	2264
EShape.GetDraggingMode	2264
EShape.GetShapeNamed	2265
EShape.HitHandle	2265
EShape.HitShape	2266
EShape.HitTest	2266
EShape.InvalidatedWorld	2267
EShape.Labeled	2267
EShape.LabeledRecursive	2268
EShape.Load	2268
EShape.LocalToSensor	2269
EShape.Mother	2269
EShape.Name	2269
EShape.NumDaughters	2270
EShape.PanX	2270
EShape.PanY	2271
EShape.Resizable	2271
EShape.ResizableRecursive	2271
EShape.Rotatable	2272
EShape.RotatableRecursive	2272
EShape.Save	2273
EShape.Selectable	2273
EShape.SelectableRecursive	2274
EShape.Selected	2274
EShape.SelectedRecursive	2274
EShape.SensorToLocal	2275
EShape.SetAllocated	2275
EShape.SetCursor	2276
EShape.SetDraggingMode	2277
EShape.SetPan	2277

EShape.SetZoom	2278
EShape.Type	2278
EShape.Visible	2279
EShape.VisibleRecursive	2279
EShape.WorldShape	2280
EShape.ZoomX	2280
EShape.ZoomY	2280
4.182. ESimpleCropper Class	2281
ESimpleCropper.Crop	2282
ESimpleCropper.ESimpleCropper	2282
ESimpleCropper.operator=	2283
ESimpleCropper.XRange	2284
ESimpleCropper.YRange	2284
ESimpleCropper.ZRange	2284
4.183. ESphericalCropper Class	2285
ESphericalCropper.Crop	2285
ESphericalCropper.ESphericalCropper	2286
ESphericalCropper.operator=	2287
4.184. EStatistics Class	2287
EStatistics.ComputeAverageMap	2288
EStatistics.ComputePixelStatistics	2290
EStatistics.ComputeStandardDeviationMap	2296
EStatistics.ComputeStatistics	2298
4.185. EStringPair Class	2304
EStringPair.EStringPair	2304
EStringPair.Key	2305
EStringPair.operator=	2306
EStringPair.Value	2306
4.186. EThreeLayersImageSegmenter Class	2306
EThreeLayersImageSegmenter.BlackLayerEncoded	2307
EThreeLayersImageSegmenter.BlackLayerIndex	2308
EThreeLayersImageSegmenter.NeutralLayerEncoded	2308
EThreeLayersImageSegmenter.NeutralLayerIndex	2309
EThreeLayersImageSegmenter.WhiteLayerEncoded	2309
EThreeLayersImageSegmenter.WhiteLayerIndex	2309
4.187. ETwoLayersImageSegmenter Class	2310
ETwoLayersImageSegmenter.BlackLayerEncoded	2311
ETwoLayersImageSegmenter.BlackLayerIndex	2311
ETwoLayersImageSegmenter.WhiteLayerEncoded	2311
ETwoLayersImageSegmenter.WhiteLayerIndex	2312
4.188. EUnwarpingLut Class	2312
EUnwarpingLut.EUnwarpingLut	2313
4.189. EVector Class	2313
EVector.Empty	2314
EVector.NumElements	2314
EVector.RemoveElement	2315
EVector.Serialize	2315
4.190. EWedge Class	2316
EWedge.Amplitude	2320
EWedge.ApexAngle	2320
EWedge.Breadth	2321
EWedge.CopyTo	2321
EWedge.Direct	2322
EWedge.EndAngle	2322

EWedge.EWedge	2323
EWedge.FullBreadth	2325
EWedge.FullCircle	2325
EWedge.GetCorners	2326
EWedge.GetEdges	2326
EWedge.GetInnerPoint	2327
EWedge.GetMidEdges	2328
EWedge.GetOuterPoint	2328
EWedge.GetPoint	2329
EWedge.InnerApex	2329
EWedge.InnerArcLength	2330
EWedge.InnerDiameter	2330
EWedge.InnerEnd	2331
EWedge.InnerOrg	2331
EWedge.InnerRadius	2331
EWedge.operator=	2332
EWedge.OrgAngle	2332
EWedge.OuterApex	2333
EWedge.OuterArcLength	2333
EWedge.OuterDiameter	2334
EWedge.OuterEnd	2334
EWedge.OuterOrg	2335
EWedge.OuterRadius	2335
EWedge.SetDiameters	2336
EWedge.SetFromCenterAndOrigin	2337
EWedge.SetFromOriginMiddleEnd	2337
EWedge.SetFromTwoPoints	2338
EWedge.SetRadii	2339
4.191. EWedgeGauge Class	2340
EWedgeGauge.Active	2346
EWedgeGauge.ActiveEdges	2347
EWedgeGauge.AddSkipRange	2347
EWedgeGauge.AverageDistance	2348
EWedgeGauge.CopyTo	2349
EWedgeGauge.Drag	2349
EWedgeGauge.Draw	2350
EWedgeGauge.DrawWithCurrentPen	2351
EWedgeGauge.EWedgeGauge	2352
EWedgeGauge.FilteringThreshold	2352
EWedgeGauge.GetMeasuredPoint	2353
EWedgeGauge.GetMinNumFitSamples	2354
EWedgeGauge.GetSampleA	2354
EWedgeGauge.GetSampleA	2355
EWedgeGauge.GetSampleR	2356
EWedgeGauge.GetSampleR	2357
EWedgeGauge.GetSkipRange	2358
EWedgeGauge.HitTest	2359
EWedgeGauge.HVConstraint	2360
EWedgeGauge.Measure	2360
EWedgeGauge.MeasuredWedge	2361
EWedgeGauge.MeasureSample	2361
EWedgeGauge.MeasureWithoutFitting	2362
EWedgeGauge.MinAmplitude	2362
EWedgeGauge.MinArea	2363
EWedgeGauge.NumFilteringPasses	2363

EWedgeGauge.NumSamples	2364
EWedgeGauge.NumSamplesA	2364
EWedgeGauge.NumSamplesA	2365
EWedgeGauge.NumSamplesR	2365
EWedgeGauge.NumSamplesR	2366
EWedgeGauge.NumSkipRanges	2366
EWedgeGauge.NumValidSamples	2366
EWedgeGauge.operator=	2367
EWedgeGauge.Plot	2367
EWedgeGauge.PlotWithCurrentPen	2369
EWedgeGauge.Process	2370
EWedgeGauge.RectangularSamplingArea	2370
EWedgeGauge.RemoveAllSkipRanges	2371
EWedgeGauge.RemoveSkipRange	2371
EWedgeGauge.SamplingStep	2372
EWedgeGauge.SetDiameters	2372
EWedgeGauge.SetFromOriginMiddleEnd	2373
EWedgeGauge.SetFromTwoPoints	2374
EWedgeGauge.SetMinNumFitSamples	2375
EWedgeGauge.SetRadii	2376
EWedgeGauge.Smoothing	2376
EWedgeGauge.Thickness	2377
EWedgeGauge.Threshold	2377
EWedgeGauge.Tolerance	2378
EWedgeGauge.TransitionChoice	2378
EWedgeGauge.TransitionIndex	2379
EWedgeGauge.TransitionType	2379
EWedgeGauge.Type	2380
EWedgeGauge.Valid	2380
EWedgeGauge.Wedge	2381
4.192. EWedgeShape Class	2381
EWedgeShape.Amplitude	2386
EWedgeShape.Angle	2387
EWedgeShape.ApexAngle	2387
EWedgeShape.Breadth	2388
EWedgeShape.Center	2388
EWedgeShape.CenterX	2389
EWedgeShape.CenterY	2389
EWedgeShape.Closest	2390
EWedgeShape.CopyTo	2390
EWedgeShape.Direct	2391
EWedgeShape.Drag	2391
EWedgeShape.Draw	2392
EWedgeShape.DrawWithCurrentPen	2393
EWedgeShape.EndAngle	2393
EWedgeShape.EWedgeShape	2394
EWedgeShape.FullBreadth	2395
EWedgeShape.FullCircle	2395
EWedgeShape.GetCorners	2395
EWedgeShape.GetEdges	2396
EWedgeShape.GetInnerPoint	2397
EWedgeShape.GetMidEdges	2397
EWedgeShape.GetOuterPoint	2398
EWedgeShape.GetPoint	2399
EWedgeShape.HitTest	2399

EWedgeShape.InnerApex	2400
EWedgeShape.InnerArcLength	2400
EWedgeShape.InnerDiameter	2400
EWedgeShape.InnerEnd	2401
EWedgeShape.InnerOrg	2401
EWedgeShape.InnerRadius	2402
EWedgeShape.operator=	2402
EWedgeShape.OrgAngle	2402
EWedgeShape.OuterApex	2403
EWedgeShape.OuterArcLength	2404
EWedgeShape.OuterDiameter	2404
EWedgeShape.OuterEnd	2405
EWedgeShape.OuterOrg	2405
EWedgeShape.OuterRadius	2406
EWedgeShape.Scale	2406
EWedgeShape.SetCenterXY	2406
EWedgeShape.SetDiameters	2407
EWedgeShape.SetFromCenterAndOrigin	2408
EWedgeShape.SetFromOriginMiddleEnd	2409
EWedgeShape.SetFromTwoPoints	2410
EWedgeShape.SetRadii	2410
EWedgeShape.Type	2411
EWedgeShape.Wedge	2411
4.193. EWorldShape Class	2412
EWorldShape.AddLandmark	2420
EWorldShape.AddPoint	2420
EWorldShape.Angle	2421
EWorldShape.AutoCalibrate	2422
EWorldShape.AutoCalibrateDotGrid	2422
EWorldShape.AutoCalibrateLandmarks	2423
EWorldShape.Calibrate	2424
EWorldShape.CalibrationModes	2425
EWorldShape.CalibrationSucceeded	2425
EWorldShape.Center	2426
EWorldShape.CenterX	2426
EWorldShape.CenterY	2427
EWorldShape.Closest	2427
EWorldShape.DisableTypeFilter	2428
EWorldShape.DistortionStrength	2428
EWorldShape.DistortionStrength2	2428
EWorldShape.Drag	2429
EWorldShape.DragLandmark	2429
EWorldShape.Draw	2430
EWorldShape.DrawCrossGrid	2431
EWorldShape.DrawCrossGridWithCurrentPen	2432
EWorldShape.DrawGrid	2433
EWorldShape.DrawGridWithCurrentPen	2434
EWorldShape.DrawLandmarks	2435
EWorldShape.DrawWithCurrentPen	2435
EWorldShape.EmptyLandmarks	2436
EWorldShape.EnableTypeFilter	2436
EWorldShape.EWorldShape	2437
EWorldShape.FieldHeight	2437
EWorldShape.FieldWidth	2438
EWorldShape.GetLandmarkElement	2439

EWorldShape.GridPointsMaxVariation	2439
EWorldShape.GridPointsMaxVariationThreshold	2440
EWorldShape.GridPointsMeanVariation	2440
EWorldShape.GridPointsMeanVariationThreshold	2441
EWorldShape.HitLandmark	2441
EWorldShape.HitLandmarks	2442
EWorldShape.HitTest	2442
EWorldShape.NumLandmarkElements	2443
EWorldShape.operator=	2443
EWorldShape.PanX	2444
EWorldShape.PanY	2444
EWorldShape.PerspectiveStrength	2445
EWorldShape.Ratio	2445
EWorldShape.RebuildGrid	2446
EWorldShape.RemoveLandmark	2447
EWorldShape.Scale	2447
EWorldShape.SensorHeight	2448
EWorldShape.SensorToWorld	2448
EWorldShape.SensorWidth	2449
EWorldShape.SetCenterXY	2449
EWorldShape.SetDistortion	2450
EWorldShape.SetFieldSize	2450
EWorldShape.SetPan	2451
EWorldShape.SetPerspective	2452
EWorldShape.SetResolution	2453
EWorldShape.SetSensor	2454
EWorldShape.SetSensorSize	2456
EWorldShape.SetSize	2456
EWorldShape.SetupUnwarp	2457
EWorldShape.SetZoom	2458
EWorldShape.TiltXAngle	2458
EWorldShape.TiltYAngle	2459
EWorldShape.Type	2460
EWorldShape.Unwarp	2460
EWorldShape.WorldToSensor	2461
EWorldShape.XResolution	2462
EWorldShape.YResolution	2462
EWorldShape.ZoomX	2463
EWorldShape.ZoomY	2463
4.194. EZMap Class	2463
EZMap.AddMetadata	2469
EZMap.Clear	2470
EZMap.Create	2470
EZMap.Draw	2471
EZMap.DrawImage	2474
EZMap.GetBufferPtr	2476
EZMap.GetCheckedBufferPtr	2477
EZMap.GetMetadata	2478
EZMap.GetResolution	2479
EZMap.GetSizeInWorld	2479
EZMap.GetWorldPositionFromPixelPosition	2480
EZMap.GetZMapPositionFromPixelPosition	2481
EZMap.Height	2481
EZMap.ImageToWorld	2482
EZMap.ImageToZMap	2482

EZMap.IsVoid	2483
EZMap.Load	2484
EZMap.LoadImage	2484
EZMap.LoadImageAndMetadata	2485
EZMap.LoadMetadata	2485
EZMap.MapToWorldMatrix	2486
EZMap.RowPitch	2486
EZMap.Save	2487
EZMap.SaveImage	2487
EZMap.SaveImageAndMetadata	2488
EZMap.SaveMetadata	2488
EZMap.Serialize	2489
EZMap.SerializeImage	2489
EZMap.SetBufferPtr	2490
EZMap.SetResolution	2491
EZMap.SetSize	2492
EZMap.Type	2493
EZMap.Width	2493
EZMap.WorldShape	2493
EZMap.WorldToImage	2494
EZMap.WorldToMapMatrix	2494
EZMap.WorldToZMap	2495
EZMap.XResolution	2496
EZMap.YResolution	2496
EZMap.ZMapToImage	2497
EZMap.ZMapToWorld	2497
EZMap.ZResolution	2498
4.195. EZMap16 Class	2499
EZMap16.AddMetadata	2506
EZMap16.AsEImage	2506
EZMap16.Clear	2507
EZMap16.ClearMetadata	2507
EZMap16.ConvertCoordinatesMapToPixel	2508
EZMap16.ConvertCoordinatesPixelToMap	2508
EZMap16.CopyMetadataTo	2509
EZMap16.DeleteMetadata	2510
EZMap16.Draw	2510
EZMap16.DrawImage	2514
EZMap16.EZMap16	2516
EZMap16.FillUndefinedPixels	2517
EZMap16.GetBufferPtr	2517
EZMap16.GetCheckedBufferPtr	2518
EZMap16.GetMetadata	2519
EZMap16.GetPixel	2519
EZMap16.GetPixelPositionFromWorldPosition	2520
EZMap16.GetResolution	2521
EZMap16.GetSizeInWorld	2521
EZMap16.GetWorldPositionFromPixelPosition	2522
EZMap16.GetZMapPositionFromPixelPosition	2523
EZMap16.GetZValue	2523
EZMap16.Height	2524
EZMap16.ImageToWorld	2524
EZMap16.ImageToZMap	2525
EZMap16.IsVoid	2526
EZMap16.Load	2526

EZMap16.LoadImage	2527
EZMap16.LoadImageAndMetadata	2527
EZMap16.LoadMetadata	2528
EZMap16.MapToWorldMatrix	2528
EZMap16.ModifyMetadata	2529
EZMap16.operator=	2529
EZMap16.RowPitch	2530
EZMap16.Save	2530
EZMap16.SaveImage	2531
EZMap16.SaveImageAndMetadata	2532
EZMap16.SaveMetadata	2532
EZMap16.Serialize	2533
EZMap16.SerializelImage	2533
EZMap16.SetBufferPtr	2534
EZMap16.SetPixel	2534
EZMap16.SetResolution	2535
EZMap16.SetSize	2536
EZMap16.SetZValue	2537
EZMap16.Type	2537
EZMap16.UndefinedValue	2538
EZMap16.Width	2538
EZMap16.WorldShape	2539
EZMap16.WorldToImage	2539
EZMap16.WorldToMapMatrix	2540
EZMap16.WorldToZMap	2540
EZMap16.XResolution	2541
EZMap16.YResolution	2541
EZMap16.ZMapToImage	2542
EZMap16.ZMapToWorld	2542
EZMap16.ZResolution	2543
4.196. EZMap32f Class	2544
EZMap32f.AddMetadata	2551
EZMap32f.AsEImage	2551
EZMap32f.Clear	2552
EZMap32f.ClearMetadata	2552
EZMap32f.ConvertCoordinatesMapToPixel	2553
EZMap32f.ConvertCoordinatesPixelToMap	2553
EZMap32f.CopyMetadataTo	2554
EZMap32f.DeleteMetadata	2555
EZMap32f.Draw	2555
EZMap32f.DrawImage	2559
EZMap32f.EZMap32f	2561
EZMap32f.FillUndefinedPixels	2562
EZMap32f.GetBufferPtr	2562
EZMap32f.GetCheckedBufferPtr	2563
EZMap32f.GetMetadata	2564
EZMap32f.GetPixel	2564
EZMap32f.GetPixelPositionFromWorldPosition	2565
EZMap32f.GetResolution	2566
EZMap32f.GetSizeInWorld	2566
EZMap32f.GetWorldPositionFromPixelPosition	2567
EZMap32f.GetZMapPositionFromPixelPosition	2568
EZMap32f.GetZValue	2568
EZMap32f.Height	2569
EZMap32f.ImageToWorld	2569

EZMap32f.ImageToZMap	2570
EZMap32f.IsVoid	2571
EZMap32f.Load	2571
EZMap32f.LoadImage	2572
EZMap32f.LoadImageAndMetadata	2572
EZMap32f.LoadMetadata	2573
EZMap32f.MapToWorldMatrix	2573
EZMap32f.ModifyMetadata	2574
EZMap32f.operator=	2574
EZMap32f.RowPitch	2575
EZMap32f.Save	2575
EZMap32f.SaveImage	2576
EZMap32f.SaveImageAndMetadata	2577
EZMap32f.SaveMetadata	2577
EZMap32f.Serialize	2578
EZMap32f.SerializelImage	2578
EZMap32f.SetBufferPtr	2579
EZMap32f.SetPixel	2579
EZMap32f.SetResolution	2580
EZMap32f.SetSize	2581
EZMap32f.SetZValue	2582
EZMap32f.Type	2582
EZMap32f.UndefinedValue	2583
EZMap32f.Width	2583
EZMap32f.WorldShape	2584
EZMap32f.WorldToImage	2584
EZMap32f.WorldToMapMatrix	2585
EZMap32f.WorldToZMap	2585
EZMap32f.XResolution	2586
EZMap32f.YResolution	2586
EZMap32f.ZMapToImage	2587
EZMap32f.ZMapToWorld	2587
EZMap32f.ZResolution	2588
4.197. EZMap8 Class	2589
EZMap8.AddMetadata	2596
EZMap8.AsEImage	2596
EZMap8.Clear	2597
EZMap8.ClearMetadata	2597
EZMap8.ConvertCoordinatesMapToPixel	2598
EZMap8.ConvertCoordinatesPixelToMap	2598
EZMap8.CopyMetadataTo	2599
EZMap8.DeleteMetadata	2600
EZMap8.Draw	2600
EZMap8.DrawImage	2604
EZMap8.EZMap8	2606
EZMap8.FillUndefinedPixels	2607
EZMap8.GetBufferPtr	2607
EZMap8.GetCheckedBufferPtr	2608
EZMap8.GetMetadata	2609
EZMap8.GetPixel	2609
EZMap8.GetPixelPositionFromWorldPosition	2610
EZMap8.GetResolution	2611
EZMap8.GetSizeInWorld	2611
EZMap8.GetWorldPositionFromPixelPosition	2612
EZMap8.GetZMapPositionFromPixelPosition	2613

EZMap8.GetZValue	2613
EZMap8.Height	2614
EZMap8.ImageToWorld	2614
EZMap8.ImageToZMap	2615
EZMap8.IsVoid	2616
EZMap8.Load	2616
EZMap8.LoadImage	2617
EZMap8.LoadImageAndMetadata	2617
EZMap8.LoadMetadata	2618
EZMap8.MapToWorldMatrix	2618
EZMap8.ModifyMetadata	2619
EZMap8.operator=	2619
EZMap8.RowPitch	2620
EZMap8.Save	2620
EZMap8.SaveImage	2621
EZMap8.SaveImageAndMetadata	2622
EZMap8.SaveMetadata	2622
EZMap8.Serialize	2623
EZMap8.SerialzeImage	2623
EZMap8.SetBufferPtr	2624
EZMap8.SetPixel	2624
EZMap8.SetResolution	2625
EZMap8.SetSize	2626
EZMap8.SetZValue	2627
EZMap8.Type	2627
EZMap8.UndefinedValue	2628
EZMap8.Width	2628
EZMap8.WorldShape	2629
EZMap8.WorldToImage	2629
EZMap8.WorldToMapMatrix	2630
EZMap8.WorldToZMap	2630
EZMap8.XResolution	2631
EZMap8.YResolution	2631
EZMap8.ZMapToImage	2632
EZMap8.ZMapToWorld	2632
EZMap8.ZResolution	2633
4.198. EZMapToPointCloudConverter Class	2633
EZMapToPointCloudConverter.Convert	2634
EZMapToPointCloudConverter.EZMapToPointCloudConverter	2636
5. Structures	2637
5.1. E3DPoint Struct	2637
E3DPoint.DistanceTo	2638
E3DPoint.E3DPoint	2638
E3DPoint.operator!=	2639
E3DPoint.operator==	2639
E3DPoint.Serialize	2640
E3DPoint.X	2640
E3DPoint.Y	2641
E3DPoint.Z	2641
5.2. EBW1 Struct	2641
EBW1.EBW1	2642
EBW1.Size	2643
EBW1.Value	2643
5.3. EBW16 Struct	2643

EBW16.EBW16	2644
EBW16.Size	2645
EBW16.Value	2645
5.4. EBW16Path Struct	2645
EBW16Path.Pixel	2646
EBW16Path.X	2646
EBW16Path.Y	2647
5.5. EBW32 Struct	2647
EBW32.EBW32	2647
EBW32.Size	2648
EBW32.Value	2648
5.6. EBW8 Struct	2649
EBW8.EBW8	2649
EBW8.Size	2650
EBW8.Value	2650
5.7. EBW8Path Struct	2651
EBW8Path.Pixel	2651
EBW8Path.X	2652
EBW8Path.Y	2652
5.8. EC15 Struct	2652
EC15.C0	2653
EC15.C1	2653
EC15.C2	2654
EC15.EC15	2654
EC15.Size	2655
5.9. EC16 Struct	2655
EC16.C0	2656
EC16.C1	2656
EC16.C2	2657
EC16.EC16	2657
EC16.Size	2658
5.10. EC24 Struct	2658
EC24.C0	2659
EC24.C1	2659
EC24.C2	2660
EC24.EC24	2660
EC24.Size	2661
5.11. EC24A Struct	2661
EC24A.A	2662
EC24A.C0	2662
EC24A.C1	2663
EC24A.C2	2663
EC24A.EC24A	2663
EC24A.Size	2664
5.12. EC24Path Struct	2665
EC24Path.Pixel	2665
EC24Path.X	2665
EC24Path.Y	2666
5.13. EC48 Struct	2666
EC48.C0	2667
EC48.C1	2667
EC48.C2	2667
EC48.EC48	2668
EC48.Size	2668

5.14. EColor Struct	2669
EColor.C0	2669
EColor.C1	2670
EColor.C2	2670
EColor.EColor	2670
5.15. EDepth16 Struct	2671
EDepth16.EDepth16	2672
EDepth16.Size	2672
EDepth16.Value	2673
5.16. EDepth32f Struct	2673
EDepth32f.EDepth32f	2673
EDepth32f.Size	2674
EDepth32f.Value	2674
5.17. EDepth8 Struct	2675
EDepth8.EDepth8	2675
EDepth8.Size	2676
EDepth8.Value	2676
5.18. EFeatureData Struct	2677
EFeatureData.FeatDataSize	2677
EFeatureData.FeatDataType	2678
EFeatureData.FeatNum	2678
EFeatureData.Size	2678
5.19. EISH Struct	2679
EISH.H	2679
EISH.I	2680
EISH.S	2680
5.20. ELAB Struct	2680
ELAB.A	2681
ELAB.B	2681
ELAB.L	2681
5.21. ELCH Struct	2682
ELCH.C	2682
ELCH.H	2683
ELCH.L	2683
5.22. ELSH Struct	2683
ELSH.H	2684
ELSH.L	2684
ELSH.S	2684
5.23. ELUV Struct	2685
ELUV.L	2685
ELUV.U	2686
ELUV.V	2686
5.24. EMatchPosition Struct	2686
EMatchPosition.Angle	2687
EMatchPosition.AreaRatio	2688
EMatchPosition.CenterX	2688
EMatchPosition.CenterY	2689
EMatchPosition.Interpolated	2689
EMatchPosition.Scale	2689
EMatchPosition.ScaleX	2690
EMatchPosition.ScaleY	2690
EMatchPosition.Score	2691
5.25. EMatrixCodelso15415GradingParameters Struct	2691
EMatrixCodelso15415GradingParameters.AxialNonUniformity	2692

EMatrixCodelso15415GradingParameters.AxialNonUniformityGrade	2693
EMatrixCodelso15415GradingParameters.DecodingGrade	2693
EMatrixCodelso15415GradingParameters.EMatrixCodelso15415GradingParameters	2694
EMatrixCodelso15415GradingParameters.FixedPatternDamageGrade	2694
EMatrixCodelso15415GradingParameters.GridNonUniformity	2694
EMatrixCodelso15415GradingParameters.GridNonUniformityGrade	2695
EMatrixCodelso15415GradingParameters.HorizontalPrintGrowth	2695
EMatrixCodelso15415GradingParameters.ModulationGrade	2695
EMatrixCodelso15415GradingParameters.OverallSymbolGrade	2696
EMatrixCodelso15415GradingParameters.ReflectanceMarginGrade	2696
EMatrixCodelso15415GradingParameters.SymbolContrast	2697
EMatrixCodelso15415GradingParameters.SymbolContrastGrade	2697
EMatrixCodelso15415GradingParameters.UnusedErrorCorrection	2697
EMatrixCodelso15415GradingParameters.UnusedErrorCorrectionGrade	2698
EMatrixCodelso15415GradingParameters.VerticalPrintGrowth	2698
5.26. EMatrixCodelso29158CalibrationParameters Struct	2699
EMatrixCodelso29158CalibrationParameters.EMatrixCodelso29158CalibrationParameters	2699
EMatrixCodelso29158CalibrationParameters.MLcal	2700
EMatrixCodelso29158CalibrationParameters.Rcal	2700
EMatrixCodelso29158CalibrationParameters.SRcal	2700
EMatrixCodelso29158CalibrationParameters.SRtarget	2701
5.27. EMatrixCodelso29158GradingParameters Struct	2701
EMatrixCodelso29158GradingParameters.CellContrastGrade	2702
EMatrixCodelso29158GradingParameters.CellModulationGrade	2702
EMatrixCodelso29158GradingParameters.EMatrixCodelso29158GradingParameters	2703
EMatrixCodelso29158GradingParameters.FixedPatternDamageGrade	2703
EMatrixCodelso29158GradingParameters.IsMeanLightInRequiredBounds	2704
EMatrixCodelso29158GradingParameters.MeanLight	2704
EMatrixCodelso29158GradingParameters.MinimumReflectanceGrade	2704
EMatrixCodelso29158GradingParameters.OverallSymbolGrade	2705
5.28. EMatrixCodeSemiT10GradingParameters Struct	2705
EMatrixCodeSemiT10GradingParameters.CellDefects	2706
EMatrixCodeSemiT10GradingParameters.DataMatrixCellHeight	2707
EMatrixCodeSemiT10GradingParameters.DataMatrixCellWidth	2707
EMatrixCodeSemiT10GradingParameters.EMatrixCodeSemiT10GradingParameters	2707
EMatrixCodeSemiT10GradingParameters.FinderPatternDefects	2708
EMatrixCodeSemiT10GradingParameters.HorizontalMarkGrowth	2708
EMatrixCodeSemiT10GradingParameters.HorizontalMarkMisplacement	2708
EMatrixCodeSemiT10GradingParameters.SymbolContrast	2709
EMatrixCodeSemiT10GradingParameters.SymbolContrastSNR	2709
EMatrixCodeSemiT10GradingParameters.UnusedErrorCorrection	2709
EMatrixCodeSemiT10GradingParameters.VerticalMarkGrowth	2710
EMatrixCodeSemiT10GradingParameters.VerticalMarkMisplacement	2710
5.29. EMatrixPosition Struct	2711
EMatrixPosition.EMatrixPosition	2711
EMatrixPosition.operator!=	2712
EMatrixPosition.operator==	2712
EMatrixPosition.X	2713
EMatrixPosition.Y	2713
5.30. EObjectData Struct	2714
EObjectData.Class	2715
EObjectData.IsHole	2715
EObjectData.IsSelected	2715
EObjectData.ObjNbHole	2716

EObjectData.ObjNbRun	2716
EObjectData.ObjNum	2716
5.31. EOCR2CharacterCandidate Struct	2717
EOCR2CharacterCandidate.Code	2717
EOCR2CharacterCandidate.EOCR2CharacterCandidate	2718
EOCR2CharacterCandidate.Score	2718
5.32. EPath Struct	2719
EPath.X	2719
EPath.Y	2720
5.33. EPeak Struct	2720
EPeak.Amplitude	2721
EPeak.Area	2721
EPeak.Center	2721
EPeak.Length	2722
EPeak.Start	2722
5.34. ERenderStyle Struct	2722
ERenderStyle.ERenderStyle	2723
ERenderStyle.fillRGB	2723
ERenderStyle.hasFill	2724
ERenderStyle.hasLine	2724
ERenderStyle.lineRGB	2725
ERenderStyle.pointRGB	2725
5.35. ERGB Struct	2726
ERGB.B	2726
ERGB.G	2726
ERGB.R	2727
5.36. ERGBColor Struct	2727
ERGBColor.Blue	2728
ERGBColor.ERGBColor	2728
ERGBColor.Green	2729
ERGBColor.Red	2729
5.37. ERun Struct	2730
ERun.ERun	2730
ERun.Length	2731
ERun.operator!=	2731
ERun.operator==	2732
ERun.OrgX	2732
ERun.Y	2733
5.38. ERunData Struct	2733
ERunData.Class	2734
ERunData.Len	2734
ERunData.ObjNum	2734
ERunData.OrgX	2735
ERunData.OrgY	2735
5.39. EVSH Struct	2736
EVSH.H	2736
EVSH.S	2736
EVSH.V	2737
5.40. EXYZ Struct	2737
EXYZ.X	2738
EXYZ.Y	2738
EXYZ.Z	2738
5.41. EYIQ Struct	2739
EYIQ.I	2739

EYIQ.Q	2740
EYIQ.Y	2740
5.42. EYSH Struct	2740
EYSH.H	2741
EYSH.S	2741
EYSH.Y	2741
5.43. EYUV Struct	2742
EYUV.U	2742
EYUV.V	2743
EYUV.Y	2743
6. Enumerations	2744
6.1. E3DObjectFeature Enum	2744
6.2. EAdaptiveThresholdMethod Enum	2745
6.3. EAlignmentPolarity Enum	2746
6.4. EAngleUnit Enum	2746
6.5. EArithmeticLogicOperation Enum	2746
6.6. EasyOCR2CharacterFilter Enum	2749
6.7. EasyOCR2CharSpacingBias Enum	2749
6.8. EasyOCR2CharWidthBias Enum	2750
6.9. EasyOCR2DrawDetectionStyle Enum	2750
6.10. EasyOCR2DrawRecognitionStyle Enum	2751
6.11. EasyOCR2DrawSegmentationStyle Enum	2752
6.12. EasyOCR2TextPolarity Enum	2752
6.13. EAxisSystemType Enum	2753
6.14. ECalibrationMode Enum	2753
6.15. ECalibrationType Enum	2754
6.16. ECannyThresholdingMode Enum	2755
6.17. ECC000Family Enum	2755
6.18. ECharCreationMode Enum	2756
6.19. EClippingMode Enum	2756
6.20. EColorQuantization Enum	2757
6.21. EColorRampMode Enum	2757
6.22. EColorSystem Enum	2758
6.23. EConnexity Enum	2759
6.24. EContourMode Enum	2759
6.25. EContourThreshold Enum	2760
6.26. ECorrelationMode Enum	2760
6.27. EDataSize Enum	2761
6.28. EDataType Enum	2761
6.29. EDegreesOfFreedom Enum	2762
6.30. EDiagnostic Enum	2762
6.31. EDoubleThresholdMode Enum	2763
6.32. EDraggingMode Enum	2763
6.33. EDragHandle Enum	2764
6.34. EDrawableFeature Enum	2766
6.35. EDrawingMode Enum	2767
6.36. EEditionMode Enum	2768
6.37. EEncodingConnexity Enum	2769
6.38. EError Enum	2769
6.39. EFamily Enum	2800

6.40. EFeature Enum	2801
6.41. EFillUndefinedPixelsDirection Enum	2806
6.42. EFillUndefinedPixelsMethod Enum	2806
6.43. EFilteringMode Enum	2807
6.44. EFindContrastMode Enum	2807
6.45. EFlipping Enum	2808
6.46. EFramePosition Enum	2808
6.47. EGrayscaleSingleThreshold Enum	2809
6.48. EHarrisThresholdingMode Enum	2810
6.49. EHistogramFeature Enum	2810
6.50. EHitAndMissValue Enum	2811
6.51. EImageFileType Enum	2811
6.52. EImageType Enum	2812
6.53. EKernelRectifier Enum	2813
6.54. EKernelRotation Enum	2814
6.55. EKernelType Enum	2814
6.56. ELearningMode Enum	2816
6.57. ELearnParam Enum	2817
6.58. ELegacyFeature Enum	2817
6.59. ELocalSearchMode Enum	2821
6.60. ELocationMode Enum	2822
6.61. ELogicalSize Enum	2823
6.62. EMailBarcodeOrientation Enum	2826
6.63. EMailBarcodeSymbologies Enum	2827
6.64. EMapConversionMode Enum	2828
6.65. EMapConversionMode Enum	2828
6.66. EMatchContrastMode Enum	2828
6.67. EMatchingMode Enum	2829
6.68. EMatrixCodeContrastMode Enum	2829
6.69. EMaximumAnalysisMode Enum	2830
6.70. ENoiseRemovalMethod Enum	2830
6.71. ENormalizationMode Enum	2831
6.72. EObjectBasedCalibrationPrecisionVsSpeedTradeOff Enum	2831
6.73. EObjectBasedCalibrationType Enum	2832
6.74. EOCR2DetectionMethod Enum	2832
6.75. EOCR2SegmentationMethod Enum	2833
6.76. EOCCClass Enum	2833
6.77. EOCCColor Enum	2836
6.78. EPatternType Enum	2836
6.79. EPickingMode Enum	2837
6.80. EPlaneCropperType Enum	2837
6.81. EPlotItem Enum	2838
6.82. EQRCodingMode Enum	2838
6.83. EQRCodingEncoding Enum	2839
6.84. EQRCodingLevel Enum	2839
6.85. EQRCodingModel Enum	2840
6.86. EQRCodingPerspectiveMode Enum	2840
6.87. EQRCodingScanPrecision Enum	2841
6.88. EQRCodingDetectionMethod Enum	2841

6.89. EQRDetectionTradeOff Enum	2842
6.90. EQualityIndicator Enum	2842
6.91. EReadMode Enum	2843
6.92. ERectangleMode Enum	2844
6.93. EReductionMode Enum	2845
6.94. EReferenceNoise Enum	2845
6.95. ERgbStandard Enum	2845
6.96. ERoiHit Enum	2846
6.97. ESegmentationMethod Enum	2847
6.98. ESegmentationMode Enum	2848
6.99. ESelectByPosition Enum	2848
6.100. ESelectionFlag Enum	2849
6.101. ESelectOption Enum	2850
6.102. ESerializerFileWriterMode Enum	2851
6.103. EShapeBehavior Enum	2851
6.104. EShapeType Enum	2852
6.105. EShiftingMode Enum	2853
6.106. ESingleThresholdMode Enum	2853
6.107. ESortDirection Enum	2854
6.108. ESortOption Enum	2854
6.109. EStockMeasurementUnit Enum	2854
6.110. ESymbologies Enum	2855
6.111. EThinStructureMode Enum	2858
6.112. EThresholdMode Enum	2859
6.113. ETransitionChoice Enum	2859
6.114. ETransitionType Enum	2860
6.115. EZMapOrientationVectorMode Enum	2861
6.116. EZMapReferencePlaneMode Enum	2861
6.117. Features Enum	2862

1. Pixel Accessors

1.1. EBW8PixelAccessor Class

-

Namespace: Euresys::Open_eVision_1_2

Methods

EBW8PixelAccessor

-

GetPixel

-

SetPixel

E-

B

W8PixelAccessor.EBW8PixelAccessor

-

Namespace: Euresys::Open_eVision_1_2

[C#]

```
void EBW8PixelAccessor(  
    Euresys.Open_eVision_1_2.EROIBW8 roi  
)
```


Parameters

roi

-

EBW8PixelAccessor.GetPixel

-

Namespace: Euresys::Open_eVision_1_2

```
[C#]  
  
byte GetPixel(  
    int x,  
    int y  
)
```

Parameters

x

-

y

-

EBW8PixelAccessor.SetPixel

-

Namespace: Euresys::Open_eVision_1_2

```
[C#]  
  
void SetPixel(  
    byte value,  
    int x,  
    int y  
)
```

Parameters

value
-
x
-
y
-

1.2. EBW16PixelAccessor Class

-

Namespace: Euresys::Open_eVision_1_2

Methods

EBW16PixelAccessor	-
GetPixel	-
SetPixel	E- B

W16PixelAccessor.EBW16PixelAccessor

-

Namespace: Euresys::Open_eVision_1_2

[C#]

```
void EBW16PixelAccessor(  
    Euresys.Open_eVision_1_2.EROIBW16 roi  
)
```

Parameters

roi

-

EBW16PixelAccessor.GetPixel

-

Namespace: Euresys::Open_eVision_1_2

```
[C#]  
ushort GetPixel(  
    int x,  
    int y  
)
```

Parameters

x

-

y

-

EBW16PixelAccessor.SetPixel

-

Namespace: Euresys::Open_eVision_1_2

```
[C#]
```

```
void SetPixel(
    ushort value,
    int x,
    int y
)
```

Parameters

- value*
-
- x*
-
- y*
-

1.3. EBW32PixelAccessor Class

-

Namespace: Euresys::Open_eVision_1_2

Methods

EBW32PixelAccessor	-
GetPixel	-
SetPixel	E-
	B

W32PixelAccessor.EBW32PixelAccessor

-

Namespace: Euresys::Open_eVision_1_2

```
[C#]
void EBW32PixelAccessor(
    Euresys.Open_eVision_1_2.EROIBW32 roi
)
```

Parameters

roi

-

EBW32PixelAccessor.GetPixel

-

Namespace: Euresys::Open_eVision_1_2

```
[C#]
int GetPixel(
    int x,
    int y
)
```

Parameters

x

-

y

-

EBW32PixelAccessor.SetPixel

-

Namespace: Euresys::Open_eVision_1_2

```
[C#]
void SetPixel(
    int value,
    int x,
    int y
)
```

Parameters

value

-

x

-

y

-

1.4. EC15PixelAccessor Class

-

Namespace: Euresys::Open_eVision_1_2

Methods

EC15PixelAccessor

-

GetPixel

-

SetPixel

-

EC15PixelAccessor.EC15PixelAccessor

-

Namespace: Euresys::Open_eVision_1_2

```
[C#]
void EC15PixelAccessor(
    Euresys.Open_eVision_1_2.EROIC15 roi
)
```

Parameters

roi

-

EC15PixelAccessor.GetPixel

-

Namespace: Euresys::Open_eVision_1_2

```
[C#]
Euresys.Open_eVision_1_2.EC15 GetPixel(
    int x,
    int y
)
```

Parameters

x

-

y

-

EC15PixelAccessor.SetPixel

-

Namespace: Euresys::Open_eVision_1_2

```
[C#]
void SetPixel(
    Euresys.Open_eVision_1_2.EC15 value,
    int x,
    int y
)
```

Parameters

value

-

x

-

y

-

1.5. EC16PixelAccessor Class

-

Namespace: Euresys::Open_eVision_1_2

Methods

EC16PixelAccessor

-

GetPixel

-

SetPixel

| E
C

16PixelAccessor.EC16PixelAccessor

-

Namespace: Euresys::Open_eVision_1_2

```
[C#]
void EC16PixelAccessor(
    Euresys.Open_eVision_1_2.EROIC16 roi
)
```

Parameters*roi*

-

EC16PixelAccessor.GetPixel

-

Namespace: Euresys::Open_eVision_1_2

```
[C#]
Euresys.Open_eVision_1_2.EC16 GetPixel(
    int x,
    int y
)
```

Parameters*x*

-

y
-

EC16PixelAccessor.SetPixel

-

Namespace: Euresys::Open_eVision_1_2

```
[C#]
void SetPixel(
    Euresys.Open_eVision_1_2.EC16 value,
    int x,
    int y
)
```

Parameters

value
-
x
-
y
-

1.6. EC24APixelAccessor Class

-

Namespace: Euresys::Open_eVision_1_2

Methods

EC24APixelAccessor

-

GetPixel

-

SetPixel

E-

C

EC24APixelAccessor.EC24APixelAccessor

-

Namespace: Euresys::Open_eVision_1_2

```
[C#]
void EC24APixelAccessor (
    Euresys.Open_eVision_1_2.EROIC24A roi
)
```

Parameters

roi

-

EC24APixelAccessor.GetPixel

-

Namespace: Euresys::Open_eVision_1_2

```
[C#]
Euresys.Open_eVision_1_2.EC24A GetPixel (
    int x,
    int y
)
```

Parameters

x
-
y
-

EC24APixelAccessor.SetPixel

-

Namespace: Euresys::Open_eVision_1_2

```
[C#]  
void SetPixel(  
    Euresys.Open_eVision_1_2.EC24A value,  
    int x,  
    int y  
)
```

Parameters

value
-
x
-
y
-

1.7. EC24PixelAccessor Class

-

Namespace: Euresys::Open_eVision_1_2

Methods

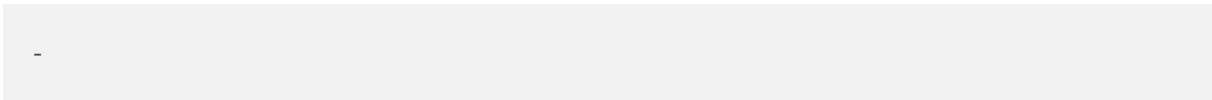
[EC24PixelAccessor](#)

GetPixel

SetPixel



24PixelAccessor.EC24PixelAccessor



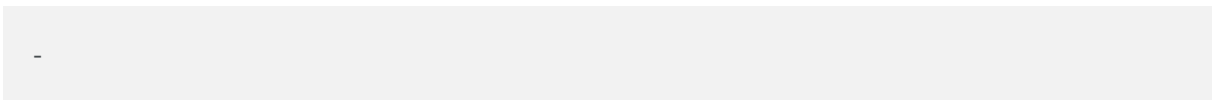
Namespace: Euresys::Open_eVision_1_2

```
[C#]
void EC24PixelAccessor(
    Euresys.Open_eVision_1_2.EROIC24 roi
)
```

Parameters

roi
-

EC24PixelAccessor.GetPixel



Namespace: Euresys::Open_eVision_1_2

```
[C#]
```

```
Euresys.Open_eVision_1_2.EC24 GetPixel(  
    int x,  
    int y  
)
```

Parameters

x

-

y

-

EC24PixelAccessor.SetPixel

-

Namespace: Euresys::Open_eVision_1_2

```
[C#]  
void SetPixel(  
    Euresys.Open_eVision_1_2.EC24 value,  
    int x,  
    int y  
)
```

Parameters

value

-

x

-

y

-

2. Common

2.1. Easy Class

Classes

"Easy Class" on page 217

2.2. Image and ROI Classes

Image Classes

"EImageBW1 Class" on page 1256
"EImageBW8 Class" on page 1263
"EImageBW16 Class" on page 1259
"EImageBW32 Class" on page 1261

"EImageC15 Class" on page 1265
"EImageC16 Class" on page 1267
"EImageC24 Class" on page 1269
"EImageC24A Class" on page 1271
"EImageC48 Class" on page 1273

ROI Classes

"EROIBW1 Class" on page 2154
"EROIBW8 Class" on page 2174
"EROIBW16 Class" on page 2161
"EROIBW32 Class" on page 2167

"EROIC15 Class" on page 2180
"EROIC16 Class" on page 2187
"EROIC24 Class" on page 2193
"EROIC24A Class" on page 2200
"EROIC48 Class" on page 2206

2.3. Region Classes

Classes

"ERectangleRegion Class" on page 2095

"EPolygonRegion Class" on page 1983

"ECircleRegion Class" on page 719

"EEllipseRegion Class" on page 1136

3. Libraries

3.1. Easy3D Library

Classes

[EAffineTransformer](#)

[E3DAxisSystem](#)

[E3DBox](#)

[ECalibrationGenerator](#)

["ECalibrationModel Class" on page 638](#)

[EConverter](#)

[EDecimator](#)

["EDepthMapToMeshConverter Class" on page 1127](#)

["EDepthMapToPointCloudConverter Class" on page 1132](#)

["EErrorStatistics Class" on page 1148](#)

["EFeaturesAligner Class" on page 1168](#)

["EFilters Class" on page 1176](#)

["EMesh Class" on page 1494](#)

["EMeshToZMapConverter Class" on page 1500](#)

[E3DPlane](#)

["EPlaneCropper Class" on page 1880](#)

["EPlaneFinder Class" on page 1885](#)

["EPlaneFitter Class" on page 1895](#)

["EPointCloud Class" on page 1914](#)

["EPointCloudFactory Class" on page 1924](#)

["EPointCloudStatistics Class" on page 1927](#)

["EPointCloudToZMapConverter Class" on page 1930](#)

["EPrincipalAxisExtractor Class" on page 1994](#)

["ERandomDecimator Class" on page 2041](#)

["ERectangularCropper Class" on page 2124](#)

["EScaleCalibrationModel Class" on page 2226](#)
["ESimpleCropper Class" on page 2281](#)
["ESphericalCropper Class" on page 2285](#)
["EStatistics Class" on page 2287](#)
[E3DViewer](#)
["EZMapToPointCloudConverter Class" on page 2633](#)

Structs

[E3DPoint](#)
[EDepth8](#)
[EDepth16](#)
[EDepth32f](#)

Enumerations

[EMaximumAnalysisMode](#)
[EAlignmentPolarity](#)
[EPlaneCropperType](#)
[EObjectBasedCalibrationType](#)
[EObjectBasedCalibrationPrecisionVsSpeedTradeOff](#)
[EZMapReferencePlaneMode](#)
[EZMapOrientationVectorMode](#)
[ENoiseRemovalMethod](#)

3.2. Easy3DLaserLine Library

Classes

["EEExplicitGeometricCalibrationModel Class" on page 1157](#)
["EObjectBasedCalibrationGenerator Class" on page 1528](#)
["EObjectBasedCalibrationModel Class" on page 1541](#)
["ELaserLineExtractor Class" on page 1317](#)

3.3. Easy3DObject Library

Classes

["E3DObject Class" on page 125](#)

["E3DObjectExtractor Class" on page 139](#)

3.4. EasyImage Library

Classes

[EasyImage](#)

[EKernel](#)

[EMovingAverage](#)

Enumerations

[EArithmeticLogicOperation](#)

[EContourMode](#)

[EContourThreshold](#)

[EHistogramFeature](#)

[EKernelRectifier](#)

[EKernelRotation](#)

[EKernelType](#)

[EReferenceNoise](#)

[ETHresholdMode](#)

3.5. EasyColor Library

Classes

EasyColor

EColorLookup

EPseudoColorLookup

Enumerations

EColorQuantization

EColorSystem

ERgbStandard

3.6. EasyObject Library

Classes

EasyObject
ECodedImage2
ECodedElement
EObject
EHole
EObjectSelection
EImageEncoder
EImageSegmenter
ETwoLayersImageSegmenter
EThreeLayersImageSegmenter
EBinaryImageSegmenter
EGrayscaleSingleThresholdSegmenter
EGrayscaleDoubleThresholdSegmenter
EColorSingleThresholdSegmenter
EColorRangeThresholdSegmenter
EImageRangeSegmenter
EReferenceImageSegmenter
ELabeledImageSegmenter
EObjectRunsIterator

Enumerations

EEncodingConnexity
ESegmentationMethod
ESingleThresholdMode
EDoubleThresholdMode
EFeature

3.7. EasyMatch Library

Classes

EMatcher

Structs

EMatchPosition

Enumerations

ECorrelationMode

EMatchContrastMode

EFilteringMode

3.8. EasyFind Library

Classes

EFoundPattern

EPatternFinder

Enumerations

EFindContrastMode

ELocalSearchMode

EPatternType

EReductionMode

EThinStructureMode

3.9. EasyGauge Library

Classes

- ECircleGauge
- ELineGauge
- EPointGauge
- ERectangleGauge
- EWedgeGauge
- EFrameShape

Enumerations

- EClippingMode
- EPlotItem
- ETransitionChoice
- ETransitionType

3.10. EasyOCR Library

Classes

- EOCR

Enumerations

- EMatchingMode
- EShiftingMode
- EOCRClass
- EOCRColor
- ESegmentationMode

3.11. EasyOCR2 Library

Classes

EOCR2
EOCR2Text
EOCR2Line
EOCR2Word
EOCR2Char

Structs

EOCR2CharacterCandidate

Enumerations

EasyOCR2CharacterFilter
EasyOCR2CharSpacingBias
EasyOCR2CharWidthBias
EasyOCR2DrawDetectionStyle
EasyOCR2DrawRecognitionStyle
EasyOCR2DrawSegmentationStyle
EasyOCR2TextPolarity

3.12. EasyOCV Library

Classes

EOCV
EOCVChar
EOCVText
EChecker

Enumerations

ECharCreationMode
EDegreesOfFreedom
EDiagnostic
ELearningMode
ELocationMode
ENormalizationMode
EQualityIndicator

3.13. EasyBarCode Library

Classes

EBarCode
EMailBarcode

Enumerations

EMailBarcodeSymbologies
EMailBarcodeOrientation

3.14. EasyMatrixCode Library

Classes

EMatrixCode
EMatrixCodeReader
ESearchParamsType

Enumerations

EFamily
EFlipping
ELearnParam
ELogicalSize
EMatrixCodeContrastMode

3.15. EasyMatrixCode2 Library

Classes

EMatrixCode2
EMatrixCode2Reader

Enumerations

"EReadMode Enum" on page 2843

3.16. EasyQRCode Library

Classes

[EQRCode](#)
[EQRCodeDecodedStream](#)
[EQRCodeDecodedStreamPart](#)
[EQRCodeGeometry](#)
[EQRCodeReader](#)
[EQuadrilateral](#)

Enumerations

[EQRCodeCodingMode](#)
[EQRCodeEncoding](#)
[EQRCodeLevel](#)
[EQRCodeModel](#)
[EQRCodeScanPrecision](#)

3.17. EasyDeepLearning Library

Classes

["EClassificationDataset Class" on page 747](#)
["EClassificationMetrics Class" on page 780](#)
["EClassificationResult Class" on page 787](#)
["EClassifier Class" on page 792](#)

3.18. Legacy

EasyObject Library (Legacy)

Classes

[ECodedImage](#)

Enumerations

[EConnexity](#)

[ELegacyFeature](#)

[ESelectByPosition](#)

[ESelectOption](#)

[ESortOption](#)

Functions

[EasyObject::ContourArea](#)

[EasyObject::ContourGravityCenter](#)

[EasyObject::ContourInertia](#)

4. Classes

4.1. E3DAxisSystem Class

Represent a 3D base axis system.

Derived Class(es): [E3DOrthonormalAxisSystem](#) [E3DRightOrthonormalAxisSystem](#)

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

AxisX	Gets the X axis.
AxisY	Gets the Y axis.
AxisZ	Gets the Z axis.
NormX	Gets the norm of the X axis.
NormY	Gets the norm of the Y axis.
NormZ	Gets the norm of the Z axis.
Origin	Gets the origin.

Methods

CheckIsNormal	check if axis system is Normed
CheckIsOrthogonal	check if axis system is Orthogonal
CheckIsRightHanded	check if axis system is Right
E3DAxisSystem	Constructs an E3DAxisSystem .
IsNormal	return true if this base axis system is Normed
IsOrthogonal	return true if this base axis system is Orthogonal
IsRightHanded	return true if this base axis system is Right Handed
operator!=	-
operator=	Assignment operator.
operator==	operator comparison
Serialize	Serializes the object with all attributes.

E3DAxisSystem.AxisX

Gets the X axis.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.Easy3D.E3DPoint AxisX  
{ get; }
```

E3DAxisSystem.AxisY

Gets the Y axis.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.Easy3D.E3DPoint AxisY  
{ get; }
```

E3DAxisSystem.AxisZ

Gets the Z axis.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
Euresys.Open_eVision_2_10.Easy3D.E3DPoint AxisZ  
  
{ get; }
```

E3DAxisSystem.CheckIsNormal

check if axis system is Normed

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
bool CheckIsNormal(  
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint axisX,  
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint axisY,  
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint axisZ  
)
```

Parameters

axisX

-

axisY

-

axisZ

-

E3DAxisSystem.CheckIsOrthogonal

check if axis system is Orthogonal

Namespace: Euresys.Open_eVision_2_10.Easy3D


```
[C#]
bool CheckIsOrthogonal(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint axisX,
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint axisY,
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint axisZ
)
```

Parameters

axisX

-

axisY

-

axisZ

-

E3DAxisSystem.CheckIsRightHanded

check if axis system is Right

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool CheckIsRightHanded(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint axisX,
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint axisY,
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint axisZ
)
```

Parameters

axisX

-

axisY

-

axisZ

E3DAxisSystem.E3DAxisSystem

Constructs an [E3DAxisSystem](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void E3DAxisSystem(
)
void E3DAxisSystem(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint Origin,
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint axisX,
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint axisY,
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint axisZ
)
void E3DAxisSystem(
    Euresys.Open_eVision_2_10.Easy3D.E3DAxisSystem other
)
```

Parameters

Origin

The origin of the axis system

axisX

The X axis

axisY

The Y axis

axisZ

The Z axis

other

Reference to another [E3DAxisSystem](#) used for the initialization.

E3DAxisSystem.IsNormal

return true if this base axis system is Normed

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
bool IsNormal (  
)
```

E3DAxisSystem.IsOrthogonal

return true if this base axis system is Orthogonal

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
bool IsOrthogonal (  
)
```

E3DAxisSystem.IsRightHanded

return true if this base axis system is Right Handed

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
bool IsRightHanded(  
)
```

E3DAxisSystem.NormX

Gets the norm of the X axis.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float NormX  
{ get; }
```

E3DAxisSystem.NormY

Gets the norm of the Y axis.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float NormY  
{ get; }
```

E3DAxisSystem.NormZ

Gets the norm of the Z axis.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float NormZ  
    { get; }
```

E3DAxisSystem.operator!=

-

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
bool operator!=(  
    Euresys.Open_eVision_2_10.Easy3D.E3DAxisSystem other  
)
```

Parameters

other

-

E3DAxisSystem.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.Easy3D.E3DAxisSystem operator=(  
    Euresys.Open_eVision_2_10.Easy3D.E3DAxisSystem other  
)
```

Parameters

other

The source [E3DAxisSystem](#).

E3DAxisSystem.operator==

operator comparison

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
bool operator==(  
    Euresys.Open_eVision_2_10.Easy3D.E3DAxisSystem other  
)
```

Parameters

other

-

E3DAxisSystem.Origin

Gets the origin.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DPoint Origin
{ get; }
```

E3DAxisSystem.Serialize

Serializes the object with all attributes.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Serialize(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is read from or write to.

4.2. E3DBox Class

A 3D box, used as bounding volume for [E3DObject](#) class. A box is defined by a center, 3 axis and 3 extent for the 3 axis. A 3D point (x,y,z) is inside the [E3DBox](#) if ... By default a [E3DBox](#) is an axis aligned unit cube, centered on the origin.

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

Axes	3D orthonormal axis system of the box.
Center	3D box center.
XAxis	X axis of the box.
XSize	Size of the 3D box along its X axis .
XYQuadrangle	2D quadrangle of the E3DBox in the XY plane
YAxis	Y axis of the box.
YSize	Size of the 3D box along its Y axis.
ZAxis	Z axis of the box.
ZSize	M Size of the 3D box along its Z axis .

e

Methods

E3DBox	Constructs an default E3DBox . By default a E3DBox is an axis aligned unit cube, centered on the origin.
--------	--

Load	Loads the E3DBox . The given ESerializer must have been created for reading.
operator!=	Checks if two E3DBox are different
operator=	Assignment operator for the E3DBox .
operator==	Checks if two E3DBox are stricly equal
Save	Saves the E3DBox . The given ESerializer must have been created for writing.

3

DBox.Axes

3D orthonormal axis system of the box.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DOrthonormalAxisSystem Axes
{ get; set; }
```

E3DBox.Center

3D box center.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DPoint Center
{ get; }
```

E3DBox.E3DBox

Constructs an default [E3DBox](#). By default a [E3DBox](#) is an axis aligned unit cube, centered on the origin.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void E3DBox(
)

void E3DBox(
    float xSize,
    float ySize,
    float zSize
)

void E3DBox(
    Euresys.Open_eVision_2_10.Easy3D.E3DOrthonormalAxisSystem axes,
    float xSize,
    float ySize,
    float zSize
)

void E3DBox(
    Euresys.Open_eVision_2_10.Easy3D.E3DBox other
)
```

```
void E3DBox(  
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint center,  
    float roll,  
    float pitch,  
    float yaw,  
    float xSize,  
    float ySize,  
    float zSize  
)
```

Parameters

xSize

The full size of the box along the X axis.

ySize

The full size of the box along the Y axis.

zSize

The full size of the box along the Z axis.

axes

Axis system.

other

Reference to another [E3DBox](#) used for the initialization.

center

The 3D coordinate of the box center.

roll

Roll (rotation along the X axis) of the box.

pitch

Pitch (rotation along the Y axis) of the box.

yaw

Yaw (rotation along the Z axis) of the box.

E3DBox.Load

Loads the [E3DBox](#). The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Load(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

-

E3DBox.operator!=

Checks if two [E3DBox](#) are different

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool operator!=(
    Euresys.Open_eVision_2_10.Easy3D.E3DBox other
)
```

Parameters

other

-

E3DBox.operator=

Assignment operator for the [E3DBox](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DBox operator=(
    Euresys.Open_eVision_2_10.Easy3D.E3DBox other
)
```

Parameters

other

-

E3DBox.operator==

Checks if two [E3DBox](#) are stricly equal

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool operator==(
    Euresys.Open_eVision_2_10.Easy3D.E3DBox other
)
```

Parameters

other

-

E3DBox.Save

Saves the [E3DBox](#). The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

-

E3DBox.XAxis

X axis of the box.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DPoint XAxis
{ get; }
```

E3DBox.XSize

Size of the 3D box along its X axis .

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
float XSize
{ get; set; }
```

E3DBox.XYQuadrangle

2D quadrangle of the [E3DBox](#) in the XY plane

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.EQuadrangle XYQuadrangle  
    { get; }
```

E3DBox.YAxis

Y axis of the box.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.Easy3D.E3DPoint YAxis  
    { get; }
```

E3DBox.YSize

Size of the 3D box along its Y axis.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
float YSize
```

```
{ get; set; }
```

E3DBox.ZAxis

Z axis of the box.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
Euresys.Open_eVision_2_10.Easy3D.E3DPoint ZAxis
```

```
{ get; }
```

E3DBox.ZSize

Size of the 3D box along its Z axis .

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
float ZSize
```

```
{ get; set; }
```


4.3. E3DObject Class

A [E3DObject](#) is a geometric description of a set of 3D points, produced by [E3DObjectExtractor](#). Several 3D features are available. All 3D features are expressed in the ZMap metric coordinate system.

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

Area	Object area in metric units.
AveragePosition	3D average position of the object.
BasePlane	Base plane. The base plane is calculated using points surrounding the object.
BaseTilt	Angle between the base plane and the vertical (Z) axis.
BoundingBox	The E3DBox (3D oriented bounding box) of the object. The bounding box is oriented in the XY plane of the ZMap space (rotation over the ZMap Z axis). The bounding box X and Y sizes are the object length and width (see E3DObject and E3DObject). The bounding box Z size is always in the ZMap Z axis direction.
Length	Length of the object in metric units. The length is the largest dimension of the object on the XY plane of the ZMap space.

LocalHeight

Local height of the object in metric units. The local height of the object is relative to the surroundings. The base plane is used as the reference for the calculation of the local height. If it is not possible to evaluate a base plane, the local height has the same value as the reference height ([E3DObject](#))

LocalTilt

Angle between the object plane and the base plane.

LocalTopPosition

3D highest position of the object relatively to the object base plane. If it is not possible to evaluate a base plane, the local top position is the reference top position ([E3DObject](#))

NumPixels

Number of ZMap pixels composing the object.

Orientation

Orientation of the object. The orientation is the angle between the object major (longest) axis and the ZMap X axis.

Plane

Plane fitted to the object 3D positions.

RectangleRegion

[ERectangleRegion](#) enclosing the object ZMap pixels.

ReferenceHeight

Reference height of the object in metric units. The reference height of the object is relative to the ZMap origin (also known as the reference plane).

ReferenceTilt

Angle between the object plane and the vertical (Z) axis.

ReferenceTopPosition

3D top position relative to the ZMap origin (this is the position with the highest Z coordinate)

Region

[ERegion](#) composed of the object ZMap pixels.

Volume

Object volume in metric units.

Width

M Width of the object in metric units. The width is the smallest dimension of the object on the XY plane of the ZMap space.

e

Methods

Draw

Draws the specified feature of the object in the given graphic context

E3DObject

Constructs an [E3DObject](#) with default (invalid) values

Load

Loads the [E3DObject](#). The given [ESerializer](#) must have been created for reading.

operator=

Assignment operator

operator==

Comparison operator

Save

Saves the [E3DObject](#). The given [ESerializer](#) must have been created for writing.

E3DObject.Area

Object area in metric units.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float Area  
    { get; }
```

E3DObject.AveragePosition

3D average position of the object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.Easy3D.E3DPoint AveragePosition  
    { get; }
```

E3DObject.BasePlane

Base plane. The base plane is calculated using points surrounding the object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DPlane BasePlane
{ get; }
```

E3DObject.BaseTilt

Angle between the base plane and the vertical (Z) axis.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
float BaseTilt
{ get; }
```

E3DObject.BoundingBox

The [E3DBox](#) (3D oriented bounding box) of the object. The bounding box is oriented in the XY plane of the ZMap space (rotation over the ZMap Z axis). The bounding box X and Y sizes are the object length and width (see [E3DObject](#) and [E3DObject](#)). The bounding box Z size is always in the ZMap Z axis direction.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DBox BoundingBox
{ get; }
```

E3DObject.Draw

Draws the specified feature of the object in the given graphic context

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.Easy3D.E3DObjectFeature feature,
    Euresys.Open_eVision_2_10.ERGBColor color,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)
```

Parameters

graphicContext

Graphic context on which to draw.

feature

The feature to draw.

color

The color in which to draw the feature (optional).

zoomX

-

zoomY

-

panX

-

panY

-

Remarks

Drawing is done in the device context associated to the desired window.

E3DObject.E3DObject

Constructs an [E3DObject](#) with default (invalid) values

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void E3DObject(
)
void E3DObject(
    Euresys.Open_eVision_2_10.Easy3D.E3DObject other
)
```

Parameters

other

-

E3DObject.Length

Length of the object in metric units. The length is the largest dimension of the object on the XY plane of the ZMap space.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
float Length
{ get; }
```

E3DObject.Load

Loads the [E3DObject](#). The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Load(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

-

E3DObject.LocalHeight

Local height of the object in metric units. The local height of the object is relative to the surroundings. The base plane is used as the reference for the calculation of the local height. If it is not possible to evaluate a base plane, the local height has the same value as the reference height ([E3DObject](#))

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
float LocalHeight
{ get; }
```


E3DObject.LocalTilt

Angle between the object plane and the base plane.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float LocalTilt  
  
{ get; }
```

E3DObject.LocalTopPosition

3D highest position of the object relatively to the object base plane. If it is not possible to evaluate a base plane, the local top position is the reference top position ([E3DObject](#))

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.Easy3D.E3DPoint LocalTopPosition  
  
{ get; }
```

E3DObject.NumPixels

Number of ZMap pixels composing the object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
int NumPixels  
{ get; }
```

E3DObject.operator=

Assignment operator

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
Euresys.Open_eVision_2_10.Easy3D.E3DObject operator=(  
    Euresys.Open_eVision_2_10.Easy3D.E3DObject other  
)
```

Parameters

other

-

E3DObject.operator==

Comparison operator

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
bool operator==(
    Euresys.Open_eVision_2_10.Easy3D.E3DObject other
)
```

Parameters

other

The other [E3DObject](#).

E3DObject.Orientation

Orientation of the object. The orientation is the angle between the object major (longest) axis and the ZMap X axis.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
float Orientation
{ get; }
```

E3DObject.Plane

Plane fitted to the object 3D positions.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DPlane Plane
{ get; }
```

E3DObject.RectangleRegion

[ERectangleRegion](#) enclosing the object ZMap pixels.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.ERectangleRegion RectangleRegion  
  
    { get; }
```

E3DObject.ReferenceHeight

Reference height of the object in metric units. The reference height of the object is relative to the ZMap origin (also known as the reference plane).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float ReferenceHeight  
  
    { get; }
```

E3DObject.ReferenceTilt

Angle between the object plane and the vertical (Z) axis.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float ReferenceTilt  
  
    { get; }
```

E3DObject.ReferenceTopPosition

3D top position relative to the ZMap origin (this is the position with the highest Z coordinate)

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.Easy3D.E3DPoint ReferenceTopPosition  
  
    { get; }
```

E3DObject.Region

[ERegion](#) composed of the object ZMap pixels.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.ERegion Region  
  
    { get; }
```

E3DObject.Save

Saves the [E3DObject](#). The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

-

E3DObject.Volume

Object volume in metric units.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
float Volume
{ get; }
```

E3DObject.Width

Width of the object in metric units. The width is the smallest dimension of the object on the XY plane of the ZMap space.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float Width  
{ get; }
```

4.4. E3DObjectExtractor Class

[E3DObjectExtractor](#) is used to extract 3D objects from a ZMap.

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

AreaRange

The allowed area range for the objects. Area is expressed in the metric coordinate system. E.g, if the ZMap space is in mm, the area is in mm².

AspectRatioRange

The extraction 2D aspect ratio range. The aspect ratio is the smallest dimension divided by the largest dimension, its value is always between 0 and 1. The smaller the ratio, the more elongated the object is. A value of 1 means a perfectly square object (length=width).

LengthRange

The extraction object length range in metric units. Length is the largest dimension of the object, expressed the ZMap coordinate system.

LocalHeightRange

The extraction object local height range in metric units. Local height is the dimension of the object along the normal of the base plane. For a height based on the ZMap origin, use [E3DObjectExtractor](#).

LocalTiltRange

The allowed angle range of the object local tilt. This is the angle between the base plane and the object plane. A value of 0 means that the object top surface is parallel to its base. Valid values are between 0 and +90 degrees (or 0 and Pi if angle unit is radians).

Objects

Returns the list of extracted objects.

OrientationRange

The allowed angle range of the oriented 2D rectangle region. This is the angle of the longest axis (the length of the object), in counter clockwise direction, from the horizontal axis. Valid values are between -90 and +90 degrees (or -Pi/2 and Pi/2 if angle unit is radians).

ReferenceHeightRange

The extraction object reference height range in metric units. Reference height is the dimension of the object along the ZMap Z axis from ZMap origin. For a height based on the object base plane, use [E3DObjectExtractor](#).

ReferenceTiltRange

The allowed angle range of the object reference tilt. This is the angle between the object plane and the ZMap Z Axis. A value of 0 means that the object top surface is parallel to the ZMap XY plane. Valid values are between 0 and +90 degrees (or 0 and Pi if angle unit is radians).

VolumeRange

The allowed volume range for the objects. Volume is expressed in the metric coordinate system. E.g, if the ZMap space is in mm, the volume is in mm³.

WidthRange

M The extraction object width range in metric units. Width is the smallest dimension of the object, expressed the ZMap coordinate system.

thods

AddToMesh

For all extracted objects, adds a 3D representation (in the form of a list of triangles) of the given feature to the given mesh. If the feature is not defined for the [E3DObject](#), this method has no effect.

Draw

Draws the specified feature of all extracted objects in the given graphic context. If the feature is not defined for the [E3DObject](#), this method has no effect.

E3DObjectExtractor

Constructs an [E3DObjectExtractor](#) with default (invalid) values

Extract

Processes the ZMap and extracts a list of 3D objects matching the criteria. Returns the number of extracted objects.

Load	Load the E3DObjectExtractor . The given ESerializer must have been created for reading.
operator=	Assignment operator
operator==	Comparison operator
Save	Save the E3DObjectExtractor . The given ESerializer must have been created for writing.

3

DObjectExtractor.AddToMesh

For all extracted objects, adds a 3D representation (in the form of a list of triangles) of the given feature to the given mesh. If the feature is not defined for the [E3DObject](#), this method has no effect.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void AddToMesh(
    Euresys.Open_eVision_2_10.Easy3D.E3DObjectFeature feature,
    Euresys.Open_eVision_2_10.Easy3D.EMesh mesh
)
```

Parameters

feature

The feature to draw, only 3D features are supported.

mesh

The mesh to add the graphics for.

E3DObjectExtractor.AreaRange

The allowed area range for the objects. Area is expressed in the metric coordinate system. E.g, if the ZMap space is in mm, the area is in mm².

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.EFloatRange AreaRange  
{ get; set; }
```

E3DObjectExtractor.AspectRatioRange

The extraction 2D aspect ratio range. The aspect ratio is the smallest dimension divided by the largest dimension, its value is always between 0 and 1. The smaller the ratio, the more elongated the object is. A value of 1 means a perfectly square object (length=width).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.EFloatRange AspectRatioRange  
{ get; set; }
```

E3DObjectExtractor.Draw

Draws the specified feature of all extracted objects in the given graphic context. If the feature is not defined for the [E3DObject](#), this method has no effect.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.Easy3D.E3DObjectFeature feature,
    Euresys.Open_eVision_2_10.ERGBColor color,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)
```

Parameters

graphicContext

Graphic context on which to draw.

feature

The feature to draw.

color

The color in which to draw the feature (optional).

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

Remarks

<Drawing is done in the device context associated to the desired window.

E3DObjectExtractor.E3DObjectExtractor

Constructs an [E3DObjectExtractor](#) with default (invalid) values

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void E3DObjectExtractor(
)
void E3DObjectExtractor(
    Euresys.Open_eVision_2_10.Easy3D.E3DObjectExtractor other
)
```

Parameters

other

-

E3DObjectExtractor.Extract

Processes the ZMap and extracts a list of 3D objects matching the criteria. Returns the number of extracted objects.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
int Extract(
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 zMap
)
int Extract(
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 zMap,
    Euresys.Open_eVision_2_10.ERegion region
)
int Extract(
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 zMap
)
```

```
int Extract(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 zMap,  
    Euresys.Open_eVision_2_10.ERegion region  
)  
  
int Extract(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap32f zMap  
)  
  
int Extract(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap32f zMap,  
    Euresys.Open_eVision_2_10.ERegion region  
)  
  
int Extract(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap zMap  
)  
  
int Extract(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap zMap,  
    Euresys.Open_eVision_2_10.ERegion region  
)
```

Parameters

zMap

The source ZMap.

region

The region of interest, only pixels inside the given region are considered for object extraction.

E3DObjectExtractor.LengthRange

The extraction object length range in metric units. Length is the largest dimension of the object, expressed the ZMap coordinate system.

Namespace: Euresys.Open_eVision_2_10.Easy3D

[C#]

```
Euresys.Open_eVision_2_10.EFloatRange LengthRange
```

```
{ get; set; }
```

E3DObjectExtractor.Load

Load the [E3DObjectExtractor](#). The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
void Load(  
    Euresys.Open_eVision_2_10.ESerializer serializer  
)
```

Parameters

serializer

-

E3DObjectExtractor.LocalHeightRange

The extraction object local height range in metric units. Local height is the dimension of the object along the normal of the base plane. For a height based on the ZMap origin, use [E3DObjectExtractor](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
Euresys.Open_eVision_2_10.EFloatRange LocalHeightRange
```

```
{ get; set; }
```

E3DObjectExtractor.LocalTiltRange

The allowed angle range of the object local tilt. This is the angle between the base plane and the object plane. A value of 0 means that the object top surface is parallel to its base. Valid values are between 0 and +90 degrees (or 0 and Pi if angle unit is radians).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.EFloatRange LocalTiltRange  
  
{ get; set; }
```

E3DObjectExtractor.Objects

Returns the list of extracted objects.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.Easy3D.E3DObject[] Objects  
  
{ get; }
```


E3DObjectExtractor.operator=

Assignment operator

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DObjectExtractor operator=(
    Euresys.Open_eVision_2_10.Easy3D.E3DObjectExtractor other
)
```

Parameters

other

-

E3DObjectExtractor.operator==

Comparison operator

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool operator==(
    Euresys.Open_eVision_2_10.Easy3D.E3DObjectExtractor other
)
```

Parameters

other

The other object.

E3DObjectExtractor.OrientationRange

The allowed angle range of the oriented 2D rectangle region. This is the angle of the longest axis (the length of the object), in counter clockwise direction, from the horizontal axis. Valid values are between -90 and +90 degrees (or -Pi/2 and Pi/2 if angle unit is radians).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
Euresys.Open_eVision_2_10.EFloatRange OrientationRange  
{ get; set; }
```

E3DObjectExtractor.ReferenceHeightRange

The extraction object reference height range in metric units. Reference height is the dimension of the object along the ZMap Z axis from ZMap origin. For a height based on the object base plane, use [E3DObjectExtractor](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
Euresys.Open_eVision_2_10.EFloatRange ReferenceHeightRange  
{ get; set; }
```

E3DObjectExtractor.ReferenceTiltRange

The allowed angle range of the object reference tilt. This is the angle between the object plane and the ZMap Z Axis. A value of 0 means that the object top surface is parallel to the ZMap XY plane. Valid values are between 0 and +90 degrees (or 0 and Pi if angle unit is radians).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.EFloatRange ReferenceTiltRange
    { get; set; }
```

E3DObjectExtractor.Save

Save the [E3DObjectExtractor](#). The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Save (
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

-

E3DObjectExtractor.VolumeRange

The allowed volume range for the objects. Volume is expressed in the metric coordinate system. E.g, if the ZMap space is in mm, the volume is in mm³.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.EFloatRange VolumeRange
{ get; set; }
```

E3DObjectExtractor.WidthRange

The extraction object width range in metric units. Width is the smallest dimension of the object, expressed the ZMap coordinate system.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.EFloatRange WidthRange
{ get; set; }
```

4.5. E3DOrthonormalAxisSystem Class

E3DOrthonormalAxisSystem is a subassembly of [E3DAxisSystem](#) with properties [Orthogonal](#) and [Normed](#)

Base Class: [E3DAxisSystem](#)

Namespace: Euresys.Open_eVision_2_10.Easy3D

Methods

[E3DOrthonormalAxisSystem](#)

Constructs an [E3DOrthonormalAxisSystem](#).

[operator=](#)

Assignment operator.

E3DOrthonormalAxisSystem.E3DOrthonormalAxisSystem

Constructs an [E3DOrthonormalAxisSystem](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void E3DOrthonormalAxisSystem(
)
void E3DOrthonormalAxisSystem(
    Euresys.Open_eVision_2_10.Easy3D.E3DAxisSystem other
)
void E3DOrthonormalAxisSystem(
    Euresys.Open_eVision_2_10.Easy3D.E3DOrthonormalAxisSystem other
)
void E3DOrthonormalAxisSystem(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint Origin,
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint axisX,
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint axisY,
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint axisZ
)
```

Parameters

other

Reference to another [E3DOrthonormalAxisSystem](#) used for the initialization.

Origin

The origin of the axis system

axisX

The X axis

axisY

The Y axis

axisZ

The Z axis

Remarks

throws an exception if the axis are not orthogonal and normed

E3DOrthonormalAxisSystem.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DOrthonormalAxisSystem operator=(
    Euresys.Open_eVision_2_10.Easy3D.E3DOrthonormalAxisSystem other
)
```

Parameters

other

The source [E3DOrthonormalAxisSystem](#).

4.6. E3DPlane Class

Represents a 3D plane.

The equation of the plane is " $n_vect \cdot (x,y,z) = signedDistance$ " where " n_vect " is the normal vector and " $signedDistance$ " is the signed distance from the origin to the plane.

The signed distance is positive when the vector binding the origin to the closest point on the plane has the same direction as " n_vect " and is negative when this vector has the opposite direction as " n_vect ".

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

[Normal](#)

Gets/Sets the normal vector of the plane.

[SignedDistanceFromOrigin](#)

Gets/Sets the signed distance between the origin and the plane.

Methods

Define

(re)Defines the plane parameters:

It is possible to use the normal and the signed distance of the plane.

In that case, it exits with an exception if the normal vector is the null vector.

It is also possible to use 3 points of the plane.

In that case, it exits with an exception if the 3 points are aligned.

DistanceTo

Returns the signed distance between the plane and a given point. A positive distance means that the vector connecting the plane to the point has the same direction as the normal while a negative distance means that it has the opposite direction.

E3DPlane

Creates an [E3DPlane](#) object.

It is possible to initialize it by specifying its normal and signed distance from the origin.

In that case, it exits with an exception if the norm of the normal is null.

It is also possible to initialize it by specifying 3 points of the plane.

In that case, it exits with an exception if the 3 points are aligned.

Load

Loads a [E3DPlane](#). The given ESerializer must have been created for reading.

operator-

Operator "-": returns a new [E3DPlane](#) translated in the inverse of the direction of the normal to the plane.

operator+

Operator "+": returns a new [E3DPlane](#) translated in the direction of the normal to the plane.

operator=

Assignment operator

ProjectPoint

Returns the position of the given point projected on the plane.

Save

Saves a [E3DPlane](#). The given [ESerializer](#) must have been created for writing.

3

DPlane.Define

(re)Defines the plane parameters:

It is possible to use the normal and the signed distance of the plane.

In that case, it exits with an exception if the normal vector is the null vector.

It is also possible to use 3 points of the plane.

In that case, it exits with an exception if the 3 points are aligned.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Define (
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint normal,
    float signedDistance
)

void Define (
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint point1,
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint point2,
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint point3
)
```


Parameters

normal

The normal vector, represented by an [E3DPoint](#).

signedDistance

The signed distance between the origin and the plane.

point1

First point.

point2

Second point.

point3

Third point.

Remarks

When we define a plane by specifying 3 points, the normal vector always points toward the positive Z.

E3DPlane.DistanceTo

Returns the signed distance between the plane and a given point.

A positive distance means that the vector connecting the plane to the point has the same direction as the normal while a negative distance means that it has the opposite direction.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
float DistanceTo(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint point
)
```

Parameters

point

The 3D point to measure the distance to.

E3DPlane.E3DPlane

Creates an [E3DPlane](#) object.

It is possible to initialize it by specifying its normal and signed distance from the origin.

In that case, it exits with an exception if the norm of the normal is null.

It is also possible to initialize it by specifying 3 points of the plane.

In that case, it exits with an exception if the 3 points are aligned.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void E3DPlane(
)
void E3DPlane(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint normal,
    float signedDistance
)
void E3DPlane(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint point1,
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint point2,
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint point3
)
void E3DPlane(
    Euresys.Open_eVision_2_10.Easy3D.E3DPlane other
)
```

Parameters

normal

-

signedDistance

-

point1

-

point2

-

point3

-
other

-

Remarks

When we define a plane by specifying 3 points, the normal vector always points toward the positive Z.

E3DPlane.Load

Loads a [E3DPlane](#). The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Load(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is read from.

E3DPlane.Normal

Gets/Sets the normal vector of the plane.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DPoint Normal
```

```
{ get; set; }
```

Remarks

Normal values will be stored normalized.

E3DPlane.operator-

Operator "-": returns a new [E3DPlane](#) translated in the inverse of the direction of the normal to the plane.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.Easy3D.E3DPlane operator-(  
    float offset  
)
```

Parameters

offset
offset value

E3DPlane.operator+

Operator "+": returns a new [E3DPlane](#) translated in the direction of the normal to the plane.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
Euresys.Open_eVision_2_10.Easy3D.E3DPlane operator+(  
    float offset  
)
```

Parameters

offset
offset value

E3DPlane.operator=

Assignment operator

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.Easy3D.E3DPlane operator=(  
    Euresys.Open_eVision_2_10.Easy3D.E3DPlane other  
)
```

Parameters

other
The [E3DPlane](#) object that should be copied.

E3DPlane.ProjectPoint

Returns the position of the given point projected on the plane.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DPoint ProjectPoint(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint point
)
```

Parameters

point

The 3D point to project on plane.

E3DPlane.Save

Saves a [E3DPlane](#). The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is written to.

E3DPlane.SignedDistanceFromOrigin

Gets/Sets the signed distance between the origin and the plane.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
float SignedDistanceFromOrigin  
{ get; set; }
```

4.7. E3DRightOrthonormalAxisSystem Class

E3DRightOrthonormalAxisSystem is a subassembly of [E3DAxisSystem](#) with properties Orthogonal and Normed and Right

Base Class: [E3DAxisSystem](#)

Namespace: Euresys.Open_eVision_2_10.Easy3D

Methods

[E3DRightOrthonormalAxisSystem](#)

Constructs an [E3DRightOrthonormalAxisSystem](#).

`operator=`

Assignment operator.

3

DRightOrthonormalAxisSystem.E3DRightOrthonormalAxisSystem

Constructs an [E3DRightOrthonormalAxisSystem](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void E3DRightOrthonormalAxisSystem(
)

void E3DRightOrthonormalAxisSystem(
    Euresys.Open_eVision_2_10.Easy3D.E3DAxisSystem other
)

void E3DRightOrthonormalAxisSystem(
    Euresys.Open_eVision_2_10.Easy3D.E3DRightOrthonormalAxisSystem other
)

void E3DRightOrthonormalAxisSystem(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint Origin,
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint axisX,
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint axisY,
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint axisZ
)
```

Parameters

other

Reference to another [E3DRightOrthonormalAxisSystem](#) used for the initialization.

Origin

The origin of the axis system

axisX

The X axis

axisY

The Y axis

axisZ

The Z axis

Remarks

throws an exception if the axis are not orthogonal and normed and right

E3DRightOrthonormalAxisSystem.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D


```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DRightOrthonormalAxisSystem operator=(
    Euresys.Open_eVision_2_10.Easy3D.E3DRightOrthonormalAxisSystem other
)
```

Parameters

other

The source [E3DRightOrthonormalAxisSystem](#).

4.8. E3DTransformMatrix Class

Represents a 3D transformation [4x4] matrix.

Namespace: Euresys.Open_eVision_2_10.Easy3D

Methods

[CreateAnisotropicScalingMatrix](#)

Creates an anisotropic scaling [E3DTransformMatrix](#).

[CreateIdentityMatrix](#)

Creates an identity (neutral) [E3DTransformMatrix](#).

[CreateIsotropicScalingMatrix](#)

Creates an isotropic scaling [E3DTransformMatrix](#).

[CreateOrthoBasis](#)

Creates a orthonormal [E3DTransformMatrix](#) basis (corresponds to a rigid transformation).
The vector e1, e2, e3 should form a right-handed orthogonal basis.

[CreateOrthographicProjectionMatrix](#)

Creates an orthographic projection [E3DTransformMatrix](#).

CreatePerspectiveProjectionMatrix

Creates a perspective projection [E3DTransformMatrix](#).

CreateRotationXMatrix

Creates a rotation [E3DTransformMatrix](#) around the X axis matrix.

CreateRotationYMatrix

Creates a rotation [E3DTransformMatrix](#) around the Y axis matrix.

CreateRotationZMatrix

Creates a rotation [E3DTransformMatrix](#) around the Z axis matrix.

CreateTranslationMatrix

Creates a translation [E3DTransformMatrix](#).

E3DTransformMatrix

Creates an [E3DTransformMatrix](#) object.

GetOrthoBasis

Gets the orthogonal basis represented by this transformation or throws an exception if it is not a rigid transformation.

GetValue

Gets a value from the [E3DTransformMatrix](#) object.

Inverse

Returns the inverted [E3DTransformMatrix](#).
An exception will be thrown if the determinant of the matrix is **0**.

IsRigid

Checks that the transformation is a rigid transformation (keep the distances and angles).

Load

Loads the [E3DTransformMatrix](#) object. The given [ESerializer](#) must have been created for reading.

operator!=

Checks if two [E3DTransformMatrix](#) objects are strictly different (binary level).

operator*

[E3DTransformMatrix](#) product. Combines the transformations of the two matrices.

operator+

[E3DTransformMatrix](#) sum. Sums the current and the given matrix, returns the result.

operator=

Assignment operator.

operator==

Checks if two [E3DTransformMatrix](#) objects are strictly equal (binary level).

Save

Saves the [E3DTransformMatrix](#) object. The given [ESerializer](#) must have been created for writing.

SetValue

Sets a value in the [E3DTransformMatrix](#) object.

Transpose

↳ Returns the transposed [E3DTransformMatrix](#).
↳ If the matrix is orthogonal (rotation only transformation), the transposed matrix is the inverse transformation.

3

DTransformMatrix.CreateAnisotropicScalingMatrix

Creates an anisotropic scaling [E3DTransformMatrix](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix CreateAnisotropicScalingMatrix(  
    float scaleX,  
    float scaleY,  
    float scaleZ  
)
```

Parameters

scaleX

Scaling factor along the X axis.

scaleY

Scaling factor along the Y axis.

scaleZ

Scaling factor along the Z axis.

E3DTransformMatrix.CreateIdentityMatrix

Creates an identity (neutral) [E3DTransformMatrix](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix CreateIdentityMatrix(  
)
```

E3DTransformMatrix.CreateIsotropicScalingMatrix

Creates an isotropic scaling [E3DTransformMatrix](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix CreateIsotropicScalingMatrix(
    float scale
)
```

Parameters

scale

Scaling factor.

E3DTransformMatrix.CreateOrthoBasis

Creates a orthonormal [E3DTransformMatrix](#) basis (corresponds to a rigid transformation).
The vector e1, e2, e3 should form a right-handed orthogonal basis.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix CreateOrthoBasis(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint e1,
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint e2,
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint e3,
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint t
)
```

Parameters

e1

Vector 1.

e2

Vector 2.

e3

Vector 3.

t

Translation.

E3DTransformMatrix.CreateOrthographicProjectionMatrix

Creates an orthographic projection [E3DTransformMatrix](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix CreateOrthographicProjectionMatrix(
    float width,
    float height
)
```

Parameters

width

Width of the viewport.

height

Height of the viewport.

E3DTransformMatrix.CreatePerspectiveProjectionMatrix

Creates a perspective projection [E3DTransformMatrix](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix CreatePerspectiveProjectionMatrix(  
    float distance,  
    float width,  
    float height  
)
```

Parameters

distance

Distance of the viewport to the origin.

width

Width of the viewport.

height

Height of the viewport.

E3DTransformMatrix.CreateRotationXMatrix

Creates a rotation [E3DTransformMatrix](#) around the X axis matrix.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix CreateRotationXMatrix(  
    float Angle  
)
```

Parameters

Angle

Rotation angle.

E3DTransformMatrix.CreateRotationYMatrix

Creates a rotation [E3DTransformMatrix](#) around the Y axis matrix.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix CreateRotationYMatrix(  
    float Angle  
)
```

Parameters

Angle
Rotation angle.

E3DTransformMatrix.CreateRotationZMatrix

Creates a rotation [E3DTransformMatrix](#) around the Z axis matrix.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix CreateRotationZMatrix(  
    float Angle  
)
```

Parameters

Angle
Rotation angle.

E3DTransformMatrix.CreateTranslationMatrix

Creates a translation [E3DTransformMatrix](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix CreateTranslationMatrix(
    float dX,
    float dY,
    float dZ
)
```

Parameters

dX

Translation along the X axis.

dY

Translation along the Y axis.

dZ

Translation along the Z axis.

E3DTransformMatrix.E3DTransformMatrix

Creates an [E3DTransformMatrix](#) object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void E3DTransformMatrix(
)
```

```
void E3DTransformMatrix(  
    Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix other  
)  
  
void E3DTransformMatrix(  
    double m00,  
    double m10,  
    double m20,  
    double m30,  
    double m01,  
    double m11,  
    double m21,  
    double m31,  
    double m02,  
    double m12,  
    double m22,  
    double m32,  
    double m03,  
    double m13,  
    double m23,  
    double m33  
)
```

Parameters

other

-

m00

-

m10

-

m20

-

m30

-

m01

-

m11

-

m21

-

m31
-
m02
-
m12
-
m22
-
m32
-
m03
-
m13
-
m23
-
m33
-

Remarks

The matrix is initialized with the given values.
By default, the matrix is initialized as an identity (neutral) matrix.
The value indices are m(column, row).

E3DTransformMatrix.GetOrthoBasis

Gets the orthogonal basis represented by this transformation or throws an exception if it is not a rigid transformation.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
void GetOrthoBasis (  
    ref Euresys.Open_eVision_2_10.Easy3D.E3DPoint e1,  
    ref Euresys.Open_eVision_2_10.Easy3D.E3DPoint e2,  
    ref Euresys.Open_eVision_2_10.Easy3D.E3DPoint e3,  
    ref Euresys.Open_eVision_2_10.Easy3D.E3DPoint t  
)
```

Parameters

e1

Vector 1.

e2

Vector 2.

e3

Vector 3.

t

Translation.

E3DTransformMatrix.GetValue

Gets a value from the [E3DTransformMatrix](#) object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
float GetValue (  
    uint column,  
    uint row  
)
```

Parameters

column

Column of the value to get, from 0 to 3.

row

Row of the value to get, from 0 to 3.

E3DTransformMatrix.Inverse

Returns the inverted [E3DTransformMatrix](#).
An exception will be thrown if the determinant of the matrix is **0**.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix Inverse(  
)
```

E3DTransformMatrix.IsRigid

Checks that the transformation is a rigid transformation (keep the distances and angles).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
bool IsRigid(  
)
```

E3DTransformMatrix.Load

Loads the [E3DTransformMatrix](#) object. The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Load(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The serializer.

E3DTransformMatrix.operator!=

Checks if two [E3DTransformMatrix](#) objects are strictly different (binary level).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool operator!=(
    Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix other
)
```

Parameters

other

The other matrix.

E3DTransformMatrix.operator*

[E3DTransformMatrix](#) product. Combines the transformations of the two matrices.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix operator*(
    Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix matrix
)

Euresys.Open_eVision_2_10.Easy3D.E3DPoint operator*(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint P
)
```

Parameters

matrix

Matrix to combine with the current matrix.

P

Point to transform with the current matrix.

E3DTransformMatrix.operator+

E3DTransformMatrix sum. Sums the current and the given matrix, returns the result.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix operator+(
    Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix matrix
)
```

Parameters

matrix

Matrix to add with the current matrix.

E3DTransformMatrix.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix operator=(
    Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix other
)
```

Parameters

other

An other [E3DTransformMatrix](#).

E3DTransformMatrix.operator==

Checks if two [E3DTransformMatrix](#) objects are stricly equals (binary level).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool operator==(
    Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix other
)
```

Parameters

other

The other matrix.

E3DTransformMatrix.Save

Saves the [E3DTransformMatrix](#) object. The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The serializer.

E3DTransformMatrix.SetValue

Sets a value in the [E3DTransformMatrix](#) object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetValue(
    uint column,
    uint row,
    float value
)
```

Parameters

column

Column of the value to set, from 0 to 3.

row

Row of the value to set, from 0 to 3.

value

Value to set.

E3DTransformMatrix.Transpose

Returns the transposed [E3DTransformMatrix](#).

If the matrix is orthogonal (rotation only transformation), the transposed matrix is the inverse transformation.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix Transpose (
)
```

4.9. E3DViewer Class

Manages a viewer window for [EPointCloud](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

[Colors](#)

Sets the colors associated to the [E3DViewer](#) object to be rendered.

[PointSize](#)

Displays size of the points, in pixels.

RenderDecimationLevel

Point cloud decimation level during render.

RenderGrid

Enables or disables the display of Grid.

RenderGridStep

Enables or disables the display of Grid value.

ViewDistance

Sets view distance.

WireframeMode

M Enables or disables the display of wireframe triangles.

e

thods

ConfigureRenderSource

Sets the 3D source to be rendered. It replaces the current object.

E3DViewer

Creates an [E3DViewer](#) object.

GenerateColors

Generates the colors associated to the [E3DViewer](#) object to be rendered.
Colors are calculated from point coordinates, several mappings are exposed in [EColorRampMode](#).

HideFeatureFor3DObject

Set the [E3DObjectFeature](#) of the registered [E3DObject](#) at the specified location idx to be not rendered.

HideFeatureForAll3DObjects

Set the [E3DObjectFeature](#) of all the registered [E3DObject](#) to be not rendered.

Register3DObjects

Set the list of [E3DObject](#) from which their features can be rendered.

RemoveCurrent3DObjects

Remove all [E3DObject](#) currently registered.

SetAutoRotate

Enables and configures the auto rotate display.

SetFeatureStyleFor3DObject

Set how the [E3DObjectFeature](#) of the registered [E3DObject](#) at the specified location idx should be rendered.

SetFeatureStyleForAll3DObjects

Set how the [E3DObjectFeature](#) of all the registered [E3DObject](#) should be rendered.

SetFocus

The viewer window takes the focus.

SetPosition

Sets the position of the 3D viewer window.

SetViewAngle

Sets view angle.

SetViewTarget

Sets view target position (by default, the camera position is 'look at center of the point cloud').

Show

Shows the viewer window.

ShowFeatureFor3DObject

Set the [E3DObjectFeature](#) of the registered [E3DObject](#) at the specified location idx to be rendered.

ShowFeatureForAll3DObjects

Set the [E3DObjectFeature](#) of all the registered [E3DObject](#) to be rendered.

StopAutoRotate

Stops the auto rotate display.

3

DViewer.Colors

Sets the colors associated to the [E3DViewer](#) object to be rendered.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.EC24[] Colors  
{ get; set; }
```

Remarks

List of colors, and one color per point.

E3DViewer.ConfigureRenderSource

Sets the 3D source to be rendered. It replaces the current object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```

void ConfigureRenderSource (
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud sourceObject,
    bool keepCurrentView
)

void ConfigureRenderSource (
    Euresys.Open_eVision_2_10.Easy3D.EMesh sourceObject,
    bool keepCurrentView
)

void ConfigureRenderSource (
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 sourceObject,
    bool keepCurrentView
)

void ConfigureRenderSource (
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 sourceObject,
    bool keepCurrentView
)

void ConfigureRenderSource (
    Euresys.Open_eVision_2_10.Easy3D.EZMap32f sourceObject,
    bool keepCurrentView
)

```

Parameters

sourceObject

A 3D source ([EPointCloud](#), [EMesh](#), [EZMap](#)) to render.

keepCurrentView

An optional boolean, use TRUE to keep the current view or FALSE to reset the view and center the new object.
The default value resets the view.

Remarks

For display performance purposes, the object geometry is copied into the viewer.

Subsequent modifications on the object will thus not be visible until a new call to [E3DViewer::ConfigureRenderSource](#) has been made.

The initial viewing position looks at the object center.

E3DViewer.E3DViewer

Creates an [E3DViewer](#) object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void E3DViewer(
    int orgX,
    int orgY,
    int width,
    int height,
    int parent
)

void E3DViewer(
    Euresys.Open_eVision_2_10.Easy3D.E3DViewer other
)
```

Parameters

orgX

X coordinate of the top left corner of the viewer window.

orgY

Y coordinate of the top left corner of the viewer window.

width

Width of the viewer window.

height

Height of the viewer window.

parent

Handle of the parent window of the viewer. If NULL, the viewer is built as a independent floating window.

other

Another [E3DViewer](#).

Remarks

The origin point (orgX, orgY) defines the offset of the top left corner of the viewer from the top left corner of its parent window client area.

If the window has no parent, it defines the offset from the top left corner of the screen.

If the parent window is too small to contain the viewer, the viewer will be cropped accordingly.

E3DViewer.GenerateColors

Generates the colors associated to the [E3DViewer](#) object to be rendered. Colors are calculated from point coordinates, several mappings are exposed in [EColorRampMode](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void GenerateColors(
    Euresys.Open_eVision_2_10.Easy3D.EColorRampMode mode
)
```

Parameters

mode

Mode used to compute colors.

Remarks

The conversion used to generate colors from cloud point coordinates.

E3DViewer.HideFeatureFor3DObject

Set the [E3DObjectFeature](#) of the registered [E3DObject](#) at the specified location *idx* to be not rendered.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void HideFeatureFor3DObject(
    int idx,
    Euresys.Open_eVision_2_10.Easy3D.E3DObjectFeature feature
)
```


Parameters

idx

the position in the list of registered [E3DObject](#)

feature

the feature

E3DViewer.HideFeatureForAll3DObjects

Set the [E3DObjectFeature](#) of all the registered [E3DObject](#) to be not rendered.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void HideFeatureForAll3DObjects (
    Euresys.Open_eVision_2_10.Easy3D.E3DObjectFeature feature
)
```

Parameters

feature

the feature

E3DViewer.PointSize

Displays size of the points, in pixels.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
int PointSize
{ get; set; }
```

Remarks

The size of the point (range value is **1** to **5** pixels, and **2** by default). This value is used only to draw an [EPointCloud](#), not for an [EMesh](#).

E3DViewer.Register3DObjects

Set the list of [E3DObject](#) from which their features can be rendered.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Register3DObjects (
    Euresys.Open_eVision_2_10.Easy3D.E3DObject[] objects
)
```

Parameters

objects

List of [E3DObject](#)

Remarks

The features that need to be visualize are set with show methods. The registered features have a default style. Previous setted styles is ignored. Remove the currently registered [E3DObject](#).

E3DViewer.RemoveCurrent3DObjects

Remove all [E3DObject](#) currently registered.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
void RemoveCurrent3DObjects (
)
```

E3DViewer.RenderDecimationLevel

Point cloud decimation level during render.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
int RenderDecimationLevel
{ get; set; }
```

Remarks

The viewer will only render one point every [Decimation Level] points (**1** by default, and need to be > 0). This decimation depends on the order of the points in the [EPointCloud](#).

E3DViewer.RenderGrid

Enables or disables the display of Grid.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool RenderGrid
{ get; set; }
```

Remarks

Display Grid with true (true by default).

E3DViewer.RenderGridStep

Enables or disables the display of Grid value.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
float RenderGridStep
{ get; set; }
```

Remarks

The unit of the grid step value is the same as the one from the [EPointCloud](#).
If value is smaller or equal to 0, Grid step will automatically be computed for 10 step on the X axe. (0 by default).

E3DViewer.SetAutoRotate

Enables and configures the auto rotate display.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetAutoRotate (
    float vx,
    float vy,
    float vz
)
```

Parameters

vx
Rotation speed around axis X.

vy

Rotation speed around axis Y.

vz

Rotation speed around axis Z.

E3DViewer.SetFeatureStyleFor3DObject

Set how the [E3DObjectFeature](#) of the registered [E3DObject](#) at the specified location *idx* should be rendered.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetFeatureStyleFor3DObject (
    int idx,
    Euresys.Open_eVision_2_10.Easy3D.ERenderStyle style,
    Euresys.Open_eVision_2_10.Easy3D.E3DObjectFeature feature
)
```

Parameters

idx

-

style

the style

feature

the feature

E3DViewer.SetFeatureStyleForAll3DObjects

Set how the [E3DObjectFeature](#) of all the registered [E3DObject](#) should be rendered.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetFeatureStyleForAll3DObjects (
    Euresys.Open_eVision_2_10.Easy3D.ERenderStyle style,
    Euresys.Open_eVision_2_10.Easy3D.E3DObjectFeature feature
)
```

Parameters

style

the style

feature

the feature

E3DViewer.SetFocus

The viewer window takes the focus.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetFocus (
)
```

E3DViewer.SetPosition

Sets the position of the 3D viewer window.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetPosition(
    int orgX,
    int orgY,
    int width,
    int height
)
```

Parameters

orgX

X coordinate of the top left corner of the viewer window.

orgY

Y coordinate of the top left corner of the viewer window.

width

Width of the viewer window.

height

Height of the viewer window.

E3DViewer.SetViewAngle

Sets view angle.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetViewAngle(
    float angleX,
    float angleY
)
```

Parameters

angleX

Rotation around the X axis.

angleY

Rotation around the Y axis.

E3DViewer.SetViewTarget

Sets view target position (by default, the camera position is 'look at center of the point cloud').

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
void SetViewTarget(  
    float targetX,  
    float targetY,  
    float targetZ  
)
```

Parameters

targetX

X axis target position.

targetY

-

targetZ

-

E3DViewer.Show

Shows the viewer window.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```



```
void Show(  
    )
```

E3DViewer.ShowFeatureFor3DObject

Set the [E3DObjectFeature](#) of the registered [E3DObject](#) at the specified location `idx` to be rendered.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void ShowFeatureFor3DObject(  
    int idx,  
    Euresys.Open_eVision_2_10.Easy3D.E3DObjectFeature feature  
    )
```

Parameters

idx

the position in the list of registered [E3DObject](#)

feature

the feature

E3DViewer.ShowFeatureForAll3DObjects

Set the [E3DObjectFeature](#) of all the registered [E3DObject](#) to be rendered.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ShowFeatureForAll3DObjects (
    Euresys.Open_eVision_2_10.Easy3D.E3DObjectFeature feature
)
```

Parameters

feature
the feature

E3DViewer.StopAutoRotate

Stops the auto rotate display.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void StopAutoRotate (
)
```

E3DViewer.ViewDistance

Sets view distance.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
float ViewDistance
{ get; set; }
```

Remarks

Distance between the point of the view and the object.

E3DViewer.WireframeMode

Enables or disables the display of wireframe triangles.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool WireframeMode
{ get; set; }
```

Remarks

Display wireframe triangles with true (false by default).

4.10. EAffineTransformer Class

Manages a 3D coordinates transformation context.

Remarks

By default, no transformation is done (identity matrix). The transformations are applied in the order in which the calls to AddTransform are done.

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

Transform

Sets the [E3DTransformMatrix](#) that will be used.

Methods

AddAnisotropicScalingTransform

Adds anisotropic scaling to the current [E3DTransformMatrix](#).

AddIsotropicScalingTransform

Adds isotropic scaling to the current [E3DTransformMatrix](#).

AddOrthographicProjectionTransform

Adds an orthographic projection to the current [E3DTransformMatrix](#).

AddPerspectiveProjectionTransform

Adds a perspective projection to the current [E3DTransformMatrix](#).

AddRotationXTransform

Adds rotation around the X axis to the current [E3DTransformMatrix](#).

AddRotationYTransform

Adds rotation around the Y axis to the current [E3DTransformMatrix](#).

AddRotationZTransform

Adds rotation around the Z axis to the current [E3DTransformMatrix](#).

AddTransform

Composes a custom transformation with the current [E3DTransformMatrix](#).

AddTranslationTransform

Adds translation to the current [E3DTransformMatrix](#).

ApplyMatrix

Applies a [E3DTransformMatrix](#) to a [EPointCloud](#) or a points list. If the second parameter is present, puts the transformed points in another point cloud or points list. With a single parameter, the transformation is performed in place.

ApplyTransform

Applies the current transformation to a [EPointCloud](#) or a points list. If the second parameter is present, puts the transformed points in another point cloud or points list. With a single parameter, transformation is performed in place.

CreateAnisotropicScalingMatrix

Creates an anisotropic scaling [E3DTransformMatrix](#).

CreateIdentityMatrix

Creates an identity (neutral) [E3DTransformMatrix](#).

CreateIsotropicScalingMatrix

Creates an isotropic scaling [E3DTransformMatrix](#).

CreateOrthographicProjectionMatrix

Creates an orthographic projection [E3DTransformMatrix](#).

CreatePerspectiveProjectionMatrix

Creates a perspective projection [E3DTransformMatrix](#).

CreateRotationXMatrix

Creates a rotation [E3DTransformMatrix](#) around the X axis.

CreateRotationYMatrix

Creates a rotation [E3DTransformMatrix](#) around the Y axis.

CreateRotationZMatrix

Creates a rotation [E3DTransformMatrix](#) around the Z axis.

CreateTranslationMatrix

Creates a translation [E3DTransformMatrix](#).

EAffineTransformer

Creates an [EAffineTransformer](#) object.

operator=

Assignment operator.

Reset

Resets the transformation [E3DTransformMatrix](#) to the identity.

A

EAffineTransformer.AddAnisotropicScalingTransform

Adds anisotropic scaling to the current [E3DTransformMatrix](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix AddAnisotropicScalingTransform(
    float scaleX,
    float scaleY,
    float scaleZ
)
```

Parameters

scaleX

Scaling factor along the X axis.

scaleY

Scaling factor along the Y axis.

scaleZ

Scaling factor along the Z axis.

EAffineTransformer.AddIsotropicScalingTransform

Adds isotropic scaling to the current [E3DTransformMatrix](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix AddIsotropicScalingTransform(
    float scale
)
```

Parameters

scale

Scaling factor.

EAffineTransformer.AddOrthographicProjectionTransform

Adds an orthographic projection to the current [E3DTransformMatrix](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix AddOrthographicProjectionTransform(
    float width,
    float height
)
```

Parameters

width

Width of the viewport.

height

Height of the viewport.

EAffineTransformer.AddPerspectiveProjectionTransform

Adds a perspective projection to the current [E3DTransformMatrix](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix AddPerspectiveProjectionTransform(  
    float distance,  
    float width,  
    float height  
)
```

Parameters

distance

Distance of the viewport to the origin.

width

Width of the viewport.

height

Height of the viewport.

EAffineTransformer.AddRotationXTransform

Adds rotation around the X axis to the current [E3DTransformMatrix](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```



```
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix AddRotationXTransform(  
    float Angle  
)
```

Parameters

Angle

Rotation angle.

EAffineTransformer.AddRotationYTransform

Adds rotation around the Y axis to the current [E3DTransformMatrix](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix AddRotationYTransform(  
    float Angle  
)
```

Parameters

Angle

Rotation angle.

EAffineTransformer.AddRotationZTransform

Adds rotation around the Z axis to the current [E3DTransformMatrix](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix AddRotationZTransform(
    float Angle
)
```

Parameters

Angle

Rotation angle.

EAffineTransformer.AddTransform

Composes a custom transformation with the current [E3DTransformMatrix](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix AddTransform(
    Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix matrix
)
```

Parameters

matrix

Transformation matrix.

EAffineTransformer.AddTranslationTransform

Adds translation to the current [E3DTransformMatrix](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix AddTranslationTransform(
    float dX,
    float dY,
    float dZ
)
```

Parameters

dX

Translation along the X axis.

dY

Translation along the Y axis.

dZ

Translation along the Z axis.

EAffineTransformer.ApplyMatrix

Applies a [E3DTransformMatrix](#) to a [EPointCloud](#) or a points list.
If the second parameter is present, puts the transformed points in another point cloud or points list.
With a single parameter, the transformation is performed in place.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ApplyMatrix(
    Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix matrix,
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud cloud,
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud transformedCloud
)

void ApplyMatrix(
    Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix matrix,
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud cloud
)
```

```
void ApplyMatrix(  
    Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix matrix,  
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint[] sourcePoints,  
    ref Euresys.Open_eVision_2_10.Easy3D.E3DPoint[] transformedPoints  
)
```

Parameters

matrix

Transformation matrix.

cloud

Cloud to transform.

transformedCloud

Transformed cloud.

sourcePoints

Points list to transform.

transformedPoints

Transformed points list.

EAffineTransformer.ApplyTransform

Applies the current transformation to a [EPointCloud](#) or a points list.
If the second parameter is present, puts the transformed points in another point cloud or points list.
With a single parameter, transformation is performed in place.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
void ApplyTransform(  
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud cloud,  
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud transformedCloud  
)
```

```
void ApplyTransform(  
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud cloud  
)  
  
void ApplyTransform(  
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint[] sourcePoints,  
    ref Euresys.Open_eVision_2_10.Easy3D.E3DPoint[] transformedPoints  
)
```

Parameters

cloud

Cloud to transform.

transformedCloud

Transformed cloud.

sourcePoints

Points list to transform.

transformedPoints

Transformed points list.

EAffineTransformer.CreateAnisotropicScalingMatrix

Creates an anisotropic scaling [E3DTransformMatrix](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix CreateAnisotropicScalingMatrix(  
    float scaleX,  
    float scaleY,  
    float scaleZ  
)
```

Parameters

scaleX

Scaling factor along the X axis.

scaleY

Scaling factor along the Y axis.

scaleZ

Scaling factor along the Z axis.

EAffineTransformer.CreateIdentityMatrix

Creates an indentity (neutral) [E3DTransformMatrix](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix CreateIdentityMatrix(
)
```

EAffineTransformer.CreateIsotropicScalingMatrix

Creates an isotropic scaling [E3DTransformMatrix](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix CreateIsotropicScalingMatrix(
    float scale
)
```

Parameters

scale

Scaling factor.

EAffineTransformer.CreateOrthographicProjectionMatrix

Creates an orthographic projection [E3DTransformMatrix](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix CreateOrthographicProjectionMatrix(
    float width,
    float height
)
```

Parameters

width

Width of the viewport.

height

Height of the viewport.

EAffineTransformer.CreatePerspectiveProjectionMatrix

Creates a perspective projection [E3DTransformMatrix](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix CreatePerspectiveProjectionMatrix(
    float distance,
    float width,
    float height
)
```

Parameters

distance

Distance of the viewport to the origin.

width

Width of the viewport.

height

Height of the viewport.

EAffineTransformer.CreateRotationXMatrix

Creates a rotation [E3DTransformMatrix](#) around the X axis.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix CreateRotationXMatrix(
    float Angle
)
```

Parameters

Angle

Rotation angle.

EAffineTransformer.CreateRotationYMatrix

Creates a rotation [E3DTransformMatrix](#) around the Y axis.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```



```
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix CreateRotationYMatrix(  
    float Angle  
)
```

Parameters

Angle

Rotation angle.

EAffineTransformer.CreateRotationZMatrix

Creates a rotation [E3DTransformMatrix](#) around the Z axis.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix CreateRotationZMatrix(  
    float Angle  
)
```

Parameters

Angle

Rotation angle.

EAffineTransformer.CreateTranslationMatrix

Creates a translation [E3DTransformMatrix](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix CreateTranslationMatrix(
    float dX,
    float dY,
    float dZ
)
```

Parameters

dX

Translation along the X axis.

dY

Translation along the Y axis.

dZ

Translation along the Z axis.

EAffineTransformer.EAffineTransformer

Creates an [EAffineTransformer](#) object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void EAffineTransformer(
)

void EAffineTransformer(
    Euresys.Open_eVision_2_10.Easy3D.EAffineTransformer other
)
```

Parameters

other

Another [EAffineTransformer](#).

EAffineTransformer.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EAffineTransformer operator=(
    Euresys.Open_eVision_2_10.Easy3D.EAffineTransformer other
)
```

Parameters

other

Another [EAffineTransformer](#).

EAffineTransformer.Reset

Resets the transformation [E3DTransformMatrix](#) to the identity.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Reset(
)
```

EAffineTransformer.Transform

Sets the [E3DTransformMatrix](#) that will be used.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix Transform  
{ get; set; }
```

4.11. EAngleRectifier Class

Namespace: Euresys.Open_eVision_2_10

Methods

Rectify

| E
A

ngleRectifier.Rectify

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
double Rectify(  
    double angle,  
    double mean  
)  
  
double Rectify(  
    double angle,  
    double mean,  
    Euresys.Open_eVision_2_10.EAngleUnit currentUnit  
)
```

Parameters

angle

-

mean

-

currentUnit

-

4.12. Easy Class

This class contains static properties and methods specific to the Easy library.

Namespace: Euresys.Open_eVision_2_10

Properties

[AngleUnit](#)

Current angular unit.

[DongleCount](#)

Get the number of available dongle on the system.

MaxNumberOfProcessingThreads

Maximum number of threads used internally by the Open eVision tools. This number cannot be higher than the number of processor cores available to the system. See [Easy::NumberOfAvailableProcessorCores](#).

NumberOfAvailableProcessorCores

Number of processor cores available to the system. This is the upper limit for the number of threads usable internally by the Open eVision tools. See [Easy::MaxNumberOfProcessingThreads](#)

Version

M Returns a pointer to a **NULL** terminated character string that contains the current version number of Open eVision.

e

thods

CheckLicense

Checks if a given license is available.

CheckLicenses

Check if at least one license is available. Otherwise, an exception is thrown.

CheckOemKey

Checks if the OEM key, if any, matches a given argument.

CloseImageGraphicContext

Releases the device context associated to an image.

FromRadians

Returns the angle, converted from radians to the current angle unit.

GetBestMatchingImageType

Returns the best matching image type for a given file on disk.

GetDongleInternalSerialNumber

Get the serial number of the selected dongle.

GetErrorText

Returns the description associated to a given error code.

OpenImageGraphicContext

Allows to draw in a gray-level or a color image, using any of the Windows device context functions (the image contents is altered, allowing destructive overlays).

Render3D

Prepares a three dimensional rendering of an image where the gray-level values are taken for altitudes.

RenderColorHistogram

Prepares a three dimensional rendering of the histogram of a color image: the pixels are drawn in the RGB space rather than in the XY plane.

Resize

Resizes an image without interpolation.

SetOemKey

Writes an OEM key value on a dongle.

StartTiming

Starts timing, using the system clock or performance counter.

StopTiming

Returns the time, in specified time units, elapsed since the last invocation of [Easy::StartTiming](#).

Terminate

Method not obligatory, but necessary for close dll cleanly.

ToRadians

Returns the angle, converted from current angle unit to radians.

TrueTimingResolution

Returns the actual resolution of the timing clock, in ticks per seconds.

a

sy.AngleUnit

Current angular unit.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
static Euresys.Open_eVision_2_10.EAngleUnit AngleUnit  
{ get; set; }
```

Remarks

All angles are computed using some angular unit, as well on input as on output. The desired unit can be changed at any time. By default, all angles are given in degrees (**0..360**).

Easy.CheckLicense

Checks if a given license is available.

Namespace: Euresys.Open_eVision_2_10

[C#]


```
bool CheckLicense (  
    Euresys.Open_eVision_2_10.LicenseFeatures.Features license  
)
```

Parameters

license

The license to check

Easy.CheckLicenses

Check if at least one license is available. Otherwise, an exception is thrown.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void CheckLicenses (  
)
```

Easy.CheckOemKey

Checks if the OEM key, if any, matches a given argument.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
bool CheckOemKey (
    char[] key,
    int index
)
```

Parameters

- key*
The expected value of the OEM key
- index*
The index of the dongle where the OEM key is expected. By default, the first dongle found is selected.

Remarks

The length of the OEM key must be exactly 8 characters.

Easy.CloseImageGraphicContext

Releases the device context associated to an image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void CloseImageGraphicContext (
    Euresys.Open_eVision_2_10.EImageBW8 pImage,
    IntPtr hDC
)

void CloseImageGraphicContext (
    Euresys.Open_eVision_2_10.EImageC24 pImage,
    IntPtr hDC
)
```

Parameters

- pImage*
Pointer to the target image (must be the same as that passed to [Easy::OpenImageGraphicContext](#)).
- hDC*

Handle to a device context that was produced by [Easy::OpenImageGraphicContext](#).

Easy.DongleCount

Get the number of available dongle on the system.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
static uint DongleCount  
{ get; }
```

Easy.FromRadians

Returns the angle, converted from radians to the current angle unit.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float FromRadians (  
  float angle  
)
```

Parameters

angle

Angle to be converted

Easy.GetBestMatchingImageType

Returns the best matching image type for a given file on disk.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EImageType GetBestMatchingImageType (
    string path
)
```

Parameters

path

The path to the file on disk.

Easy.GetDongleInternalSerialNumber

Get the serial number of the selected dongle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
string GetDongleInternalSerialNumber (
    int index
)
```

Parameters

index

The index of the dongle.

Easy.GetErrorText

Returns the description associated to a given error code.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
string GetErrorText(
    Euresys.Open_eVision_2_10.EError error
)
```

Parameters

error

Error code.

Easy.MaxNumberOfProcessingThreads

Maximum number of threads used internally by the Open eVision tools. This number cannot be higher than the number of processor cores available to the system. See [Easy::NumberOfAvailableProcessorCores](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
static int MaxNumberOfProcessingThreads
{ get; set; }
```

Easy.NumberOfAvailableProcessorCores

Number of processor cores available to the system. This is the upper limit for the number of threads usable internally by the Open eVision tools. See [Easy::MaxNumberOfProcessingThreads](#)

Namespace: Euresys.Open_eVision_2_10

```
[C#]
static int NumberOfAvailableProcessorCores
{ get; }
```

Easy.OpenImageGraphicContext

Allows to draw in a gray-level or a color image, using any of the Windows device context functions (the image contents is altered, allowing destructive overlays).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
IntPtr OpenImageGraphicContext (
    Euresys.Open_eVision_2_10.EImageBW8 pImage
)
IntPtr OpenImageGraphicContext (
    Euresys.Open_eVision_2_10.EImageC24 pImage
)
```

Parameters

pImage

Pointer to the target image.

Remarks

The function returns a handle to a device context associated to the image pixel data. When the device context is no more needed, call the [Easy::CloseImageGraphicContext](#) function with the same argument.

Easy.Render3D

Prepares a three dimensional rendering of an image where the gray-level values are taken for altitudes.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Render3D(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    float phi,
    float psi,
    float xScale,
    float yScale,
    float zScale,
    int dotSize
)

void Render3D(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 zImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    float phi,
    float psi,
    float xScale,
    float yScale,
    float zScale,
    int dotSize
)
```

Parameters

sourceImage

Pointer to the source image.

destinationImage

Pointer to the destination image.

phi

Rotation angle about the X-axis.

psi

Rotation angle about the Y-axis.

xScale

Magnification factor along X (should remain close to **1**).

yScale

Magnification factor along Y (should remain close to **1**).

zScale

Magnification factor along Z (should remain close to **1**).

dotSize

Size of the rendered dots; allowed values are **1, 4, 5** or **9**.

zImage

Pointer to the altitude image.

Remarks

The image is viewed by rotating it about the X-axis, then about the Y-axis. Magnification factors in the three directions (X = width, Y = height, and Z = depth) can be given.

The rendered image appears as independent dots. The dot size can be adjusted so that the surface appears more or less opaque.

The function does not display the rendered image by itself. Rather, it prepares a destination image to be displayed.

Easy.RenderColorHistogram

Prepares a three dimensional rendering of the histogram of a color image: the pixels are drawn in the RGB space rather than in the XY plane.

Namespace: Euresys.Open_eVision_2_10

[C#]


```

void RenderColorHistogram(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    float phi,
    float psi,
    float xScale,
    float yScale,
    float zScale
)

void RenderColorHistogram(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 sysImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    float phi,
    float psi,
    float xScale,
    float yScale,
    float zScale
)

```

Parameters

sourceImage

Pointer to the raw source image.

destinationImage

Pointer to the destination image.

phi

Rotation angle about the X-axis.

psi

Rotation angle about the Y-axis.

xScale

Magnification factor along X (should remain close to **1**).

yScale

Magnification factor along Y (should remain close to **1**).

zScale

Magnification factor along Z (should remain close to **1**).

sysImage

Pointer to the source image transformed into another color system.

Remarks

This allows to observe the clustering and dispersion of the RGB values.

The image is viewed by rotating it about the X-axis, then about the Y-axis. Magnification factors in the three directions (X = Red, Y = Green, and Z = Blue) can be given.

In a more advanced version, prepares a three dimensional rendering of the pixels in another system than RGB (EasyColor provides conversion means). However, the raw RGB image must still be provided to allow the display of the pixels in their usual colors.

The rendered image appears as independent dots.

The function does not display the rendered image by itself. Rather, it prepares a destination image to be displayed.

Easy.Resize

Resizes an image without interpolation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void Resize (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage
)

void Resize (
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage
)

void Resize (
    Euresys.Open_eVision_2_10.EROIC15 sourceImage,
    Euresys.Open_eVision_2_10.EROIC15 destinationImage
)

void Resize (
    Euresys.Open_eVision_2_10.EROIC16 sourceImage,
    Euresys.Open_eVision_2_10.EROIC16 destinationImage
)

void Resize (
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage
)
```

```
void Resize (
    Euresys.Open_eVision_2_10.EROIC24A sourceImage,
    Euresys.Open_eVision_2_10.EROIC24A destinationImage
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI.

Easy.SetOemKey

Writes an OEM key value on a dongle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetOemKey (
    char[] key,
    int index
)
```

Parameters

key

The OEM key value to write

index

The index of the dongle where the OEM key must be written. By default, the first dongle found is selected.

Remarks

The length of the OEM key must be exactly 8 characters. This method raises an [CannotWriteOEMKey](#) error if the value cannot be set properly.

Easy.StartTiming

Starts timing, using the system clock or performance counter.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void StartTiming(  
)
```

Easy.StopTiming

Returns the time, in specified time units, elapsed since the last invocation of [Easy::StartTiming](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int StopTiming(  
    int resolution  
)
```

Parameters

resolution

Temporal resolution, in ticks per second.

Easy.Terminate

Method not obligatory, but necessary for close dll cleanly.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Terminate(  
)
```

Easy.ToRadians

Returns the angle, converted from current angle unit to radians.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ToRadians(  
    float angle  
)
```

Parameters

angle

Angle to be converted.

Easy.TrueTimingResolution

Returns the actual resolution of the timing clock, in ticks per seconds.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int TrueTimingResolution(
)
```

Remarks

Timing granularity is hardware-dependent, but is usually better than 1 Åµs.
This function can be used to select an appropriate timing resolution when using [Easy::StopTiming](#).

Easy.Version

Returns a pointer to a **NULL** terminated character string that contains the current version number of Open eVision.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
static string Version
{ get; }
```

4.13. EasyColor Class

This class contains static properties and methods specific to the EasyColor library.

Namespace: Euresys.Open_eVision_2_10

Properties

CieAB

CIE AB white illuminant

CieAG	CIE AG white illuminant
CieAR	CIE AR white illuminant
CieD50B	CIE D50B white illuminant
CieD50G	CIE D50G white illuminant
CieD50R	CIE D50R white illuminant
CieD55B	CIE D55B white illuminant
CieD55G	CIE D55G white illuminant
CieD55R	CIE D55R white illuminant
CieD65B	CIE D65B white illuminant
CieD65G	CIE D65G white illuminant
CieD65R	CIE D65R white illuminant
CieFB	CIE FB white illuminant

CieFG	CIE FG white illuminant
CieFR	CIE FR white illuminant
CompensateNtscGamma	NTSC inverse gamma exponent
CompensatePalGamma	PAL inverse gamma exponent
CompensateSmpteGamma	NTSC inverse gamma exponent
DstQuantization	Quantization mode for output values.
NtscGamma	NTSC gamma exponent
PalGamma	PAL gamma exponent
RgbStandard	RGB definition to be used when converting between RGB and other color systems.
SmpteGamma	SMPTE gamma exponent
SrcQuantization	Quantization mode for input values.

Methods

AlphaBlend

Draws an image over an other.

AssignNearestClass

Assigns to every pixel of the source image the nearest class index *plus one* and stores its value in the destination image.

AssignNearestClassCenter

Assigns to every pixel of the source image the nearest class center and stores its value in the destination image.

BayerToC24

Converts a Bayer pattern encoded image into a color image.

C24ToBayer

Converts a color image into a Bayer pattern encoded image.

ClassAverages

Computes the average source pixel colors for every class separately.

ClassVariances

Computes the averages and variances of the image pixel colors for every class separately.

Compose

Combines three gray-level images, considered as three color planes, into a color image.

Decompose

Extracts the three color planes, considered as gray-level images, from a color image.

Dequantize

Convert a quantized value to a unquantized value of a given color system.

Format422To444

Converts a YUV 4:2:2 image to a YUV 4:4:4 image using interpolation.

Format444To422

Converts a YUV 4:4:4 image to a YUV 4:2:2 image using filtering

GetComponent

Extracts one color plane, considered as a gray-level image, from a color image.

ImproveClassCenters

Redefines the class centers by computing the average color value of the pixels assigned to each class in the source image.

IshToRgb

Convert a color from any system to RGB.

LabToRgb

Convert a color from any system to RGB.

LabToXyz

Convert a color from one system to another, including the xyz variant (reduced XYZ).

LchToRgb

Convert a color from any system to RGB.

LshToRgb

Convert a color from any system to RGB.

LuvToRgb

Convert a color from any system to RGB.

LuvToXyz

Convert a color from one system to another, including the xyz variant (reduced XYZ).

PseudoColor

Applies the mapping defined by the pseudo-color lookup table to transform a gray-level image into a color image.

Quantize

Convert an unquantized color of a given color system to a quantized color.

RegisterPlanes

Sets a color plane of a color image by using a gray-level image as component.

RgbToIsh

Convert a color from RGB to another system.

RgbToLab

Convert a color from RGB to another system.

RgbToLch

Convert a color from RGB to another system.

RgbToLsh

Convert a color from RGB to another system.

RgbToLuv

Convert a color from RGB to another system.

RgbToReducedXyz

Convert a color from one system to another, including the xyz variant (reduced XYZ).

RgbToVsh

Convert a color from RGB to another system.

RgbToXyz

Convert a color from RGB to another system.

RgbToYiq

Convert a color from RGB to another system.

RgbToYsh

Convert a color from RGB to another system.

RgbToYuv

Convert a color from RGB to another system.

SetComponent

Sets a color plane of a color image by using a gray-level image as component.

Transform

Applies a color transformation to a specified image.

TransformBayer

Converts an image, using the transformation defined by a color lookup.

VshToRgb

Convert a color from any system to RGB.

XyzToLab

Convert a color from one system to another, including the xyz variant (reduced XYZ).

XyzToLuv

Convert a color from one system to another, including the xyz variant (reduced XYZ).

XyzToRgb

Convert a color from any system to RGB.

YiqToRgb

Convert a color from any system to RGB.

YshToRgb

Convert a color from any system to RGB.

YuvToRgb

Convert a color from any system to RGB.

a

syColor.AlphaBlend

Draws an image over an other.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AlphaBlend(
    Euresys.Open_eVision_2_10.EROIC24 source,
    Euresys.Open_eVision_2_10.EROIC24 destination,
    double opacity
)
```

Parameters

source

Foreground image.

destination

Background image.

opacity

Opacity of the foreground image.

EasyColor.AssignNearestClass

Assigns to every pixel of the source image the nearest class index *plus one* and stores its value in the destination image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AssignNearestClass(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    Euresys.Open_eVision_2_10.EC24Vector classCenters
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination gray-level image/ROI.

classCenters

Pointer to the vector of the class centers.

Remarks

This generates a labeled gray-level image for use with EasyObject (see [EImageEncoder](#) and [ELabeledImageSegmenter](#)).

Note. The class index plus one is stored instead of the class index because EasyObject will never code class 0 objects. Color image segmentation allows you to decompose a color image in different regions by assigning a "class" (integer index) to every pixel. The nearest neighbor method is used, i.e. for each class a representative center is specified, and a given pixel is associated to the class with the closest center.

Color image segmentation can be used in conjunction with EasyObject to perform blob analysis on the segmented regions.

To use the color segmentation functions, the set of class centers must be specified as a vector of [EC24](#) elements. In this sense, the method is termed supervised clustering. A good way to obtain these values is to compute the average color in an ROI.

Unsupervised clustering is also made available by implementing the so called K-means method that automatically improves an initial choice of class centers.

EasyColor.AssignNearestClassCenter

Assigns to every pixel of the source image the nearest class center and stores its value in the destination image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AssignNearestClassCenter (
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    Euresys.Open_eVision_2_10.EC24Vector classCenters
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI.

classCenters

Pointer to the vector of the class centers.

Remarks

This generates a labeled color image.

Color image segmentation allows you to decompose a color image in different regions by assigning a "class" (integer index) to every pixel. The nearest neighbor method is used, i.e. for each class a representative center is specified, and a given pixel is associated to the class with the closest center.

Color image segmentation can be used in conjunction with EasyObject to perform blob analysis on the segmented regions.

To use the color segmentation functions, the set of class centers must be specified as a vector of [EC24](#) elements. In this sense, the method is termed supervised clustering. A good way to obtain these values is to compute the average color in an ROI.

Unsupervised clustering is also made available by implementing the so called K-means method that automatically improves an initial choice of class centers.

EasyColor.BayerToC24

Converts a Bayer pattern encoded image into a color image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void BayerToC24 (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    bool evenCol,
    bool evenRow,
    bool interpolate,
    bool improved
)
```

Parameters

sourceImage

Pointer to the Bayer pattern input image/ROI, stored using the 8 bits per pixel format.

destinationImage

Pointer to the color output image/ROI.

evenCol

TRUE if the leftmost image column contains no blue pixels.

evenRow

TRUE if the topmost image row contains no red pixels.

interpolate

Interpolation mode to be used for pixel reconstruction. When **FALSE**, the missing color components are merely copied from northern/western pixels; when **TRUE**, they are computed by averaging from relevant neighbors. By default, interpolation is used.

improved

Provides an access to an improved interpolation mode that reduces visible artifacts along object edges. The running time of the improved method is longer. By default, it is not used.

Remarks

The pattern can be shifted by one pixel horizontally and vertically as needed to deal with a non standard pattern origin. See also Bayer Filter.

EasyColor.C24ToBayer

Converts a color image into a Bayer pattern encoded image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void C24ToBayer(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    bool evenCol,
    bool evenRow
)
```

Parameters

sourceImage

Pointer to the color input image/ROI.

destinationImage

Pointer to the Bayer pattern output image/ROI, stored using the 8 bits per pixel format.

evenCol

TRUE if the leftmost destination image column can't contain blue pixels.

evenRow

TRUE if the topmost destination image row can't contain red pixels.

Remarks

The pattern can be shifted by one pixel horizontally and vertically as needed to deal with a non standard pattern origin. See also Bayer Filter.

EasyColor.CieAB

CIE AB white illuminant

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
static float CieAB  
    { get; }
```

EasyColor.CieAG

CIE AG white illuminant

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
static float CieAG  
    { get; }
```

EasyColor.CieAR

CIE AR white illuminant

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
static float CieAR  
  
    { get; }
```

EasyColor.CieD50B

CIE D50B white illuminant

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
static float CieD50B  
  
    { get; }
```

EasyColor.CieD50G

CIE D50G white illuminant

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
static float CieD50G  
  
    { get; }
```

EasyColor.CieD50R

CIE D50R white illuminant

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
static float CieD50R  
    { get; }
```

EasyColor.CieD55B

CIE D55B white illuminant

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
static float CieD55B  
    { get; }
```

EasyColor.CieD55G

CIE D55G white illuminant

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
static float CieD55G  
{ get; }
```

EasyColor.CieD55R

CIE D55R white illuminant

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
static float CieD55R  
{ get; }
```

EasyColor.CieD65B

CIE D65B white illuminant

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
static float CieD65B  
{ get; }
```

EasyColor.CieD65G

CIE D65G white illuminant

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
static float CieD65G  
    { get; }
```

EasyColor.CieD65R

CIE D65R white illuminant

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
static float CieD65R  
    { get; }
```

EasyColor.CieFB

CIE FB white illuminant

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
static float CieFB  
  
    { get; }
```

EasyColor.CieFG

CIE FG white illuminant

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
static float CieFG  
  
    { get; }
```

EasyColor.CieFR

CIE FR white illuminant

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
static float CieFR  
  
    { get; }
```

EasyColor.ClassAverages

Computes the average source pixel colors for every class separately.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ClassAverages (
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EC24Vector classCenters,
    Euresys.Open_eVision_2_10.EColorVector averages
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

classCenters

Pointer to the vector of the class centers.

averages

Pointer to the vector of the average color values.

Remarks

This allows measuring the actual average color of the segmented regions.

Color image segmentation allows you to decompose a color image in different regions by assigning a "class" (integer index) to every pixel. The nearest neighbor method is used, i.e. for each class a representative center is specified, and a given pixel is associated to the class with the closest center.

Color image segmentation can be used in conjunction with EasyObject to perform blob analysis on the segmented regions.

To the color segmentation functions, the set of class centers must be specified as a vector of [EC24](#) elements. In this sense, the method is termed supervised clustering. A good way to obtain these values is to compute the average color in an ROI.

Unsupervised clustering is also made available by implementing the so called K-means method that automatically improves an initial choice of class centers.

EasyColor.ClassVariances

Computes the averages and variances of the image pixel colors for every class separately.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ClassVariances (
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EC24Vector classCenters,
    Euresys.Open_eVision_2_10.EColorVector averages,
    Euresys.Open_eVision_2_10.EColorVector variances
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

classCenters

Pointer to the vector of the class centers.

averages

Pointer to the vector of the average color values.

variances

Pointer to the vector of the variance color values.

Remarks

This allows quantifying the homogeneity of the segmented regions.

Color image segmentation allows you to decompose a color image in different regions by assigning a "class" (integer index) to every pixel. The nearest neighbor method is used, i.e. for each class a representative center is specified, and a given pixel is associated to the class with the closest center. Color image segmentation can be used in conjunction with EasyObject to perform blob analysis on the segmented regions.

To the color segmentation functions, the set of class centers must be specified as a vector of [EC24](#) elements. In this sense, the method is termed supervised clustering. A good way to obtain these values is to compute the average color in an ROI.

Unsupervised clustering is also made available by implementing the so called K-means method that automatically improves an initial choice of class centers.

EasyColor.CompensateNtscGamma

NTSC inverse gamma exponent

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
static float CompensateNtscGamma  
    { get; }
```

EasyColor.CompensatePalGamma

PAL inverse gamma exponent

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
static float CompensatePalGamma  
    { get; }
```

EasyColor.CompensateSmpteGamma

NTSC inverse gamma exponent

Namespace: Euresys.Open_eVision_2_10

```
[C#]
static float CompensateSmpteGamma
{ get; }
```

EasyColor.Compose

Combines three gray-level images, considered as three color planes, into a color image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Compose(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImageOfColor0,
    Euresys.Open_eVision_2_10.EROIBW8 sourceImageOfColor1,
    Euresys.Open_eVision_2_10.EROIBW8 sourceImageOfColor2,
    Euresys.Open_eVision_2_10.EROIC24 colorDestinationImage,
    Euresys.Open_eVision_2_10.EColorLookup lookup
)

void Compose(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImageOfColor0,
    Euresys.Open_eVision_2_10.EROIBW16 sourceImageOfColor1,
    Euresys.Open_eVision_2_10.EROIBW16 sourceImageOfColor2,
    Euresys.Open_eVision_2_10.EROIC48 colorDestinationImage
)
```

Parameters

sourceImageOfColor0

Pointers to the three input gray-level component images/ROIs.

sourceImageOfColor1

Pointers to the three input gray-level component images/ROIs.

sourceImageOfColor2

Pointers to the three input gray-level component images/ROIs.

colorDestinationImage

Pointer to the output color image/ROI.

lookup

Pointer to the color lookup table, or **NULL**.

Remarks

If a color lookup is used, the resulting image undergoes the corresponding color transform. This way, it is possible to build an RGB image from the color planes of another system, or conversely.

A color image can be seen as a set of three color planes, each corresponding to a color component. The color planes are themselves continuous tone images. An [EImageC24](#) contains three [EImageBW8](#).

EasyColor.Decompose

Extracts the three color planes, considered as gray-level images, from a color image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Decompose (
    Euresys.Open_eVision_2_10.EROIC24 colorSourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImageOfColor0,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImageOfColor1,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImageOfColor2,
    Euresys.Open_eVision_2_10.EColorLookup lookup
)

void Decompose (
    Euresys.Open_eVision_2_10.EROIC48 colorSourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImageOfColor0,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImageOfColor1,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImageOfColor2
)
```

Parameters

colorSourceImage

Pointer to the input color image/ROI.

destinationImageOfColor0

Pointers to the three output gray level component images/ROIs.

destinationImageOfColor1

Pointers to the three output gray level component images/ROIs.

destinationImageOfColor2

Pointers to the three output gray level component images/ROIs.

lookup

Pointer to the color lookup table, or **NULL**.

Remarks

If a color lookup is used, the source image undergoes the corresponding color transform. This way, it is possible to get the RGB components from an image of another system, of conversely.

A color image can be seen as a set of three color planes, each corresponding to a color component. The color planes are themselves continuous tone images. An [EImageC24](#) contains three [EImageBW8](#).

EasyColor.Dequantize

Convert a quantized value to a unquantized value of a given color system.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Dequantize (
    Euresys.Open_eVision_2_10.EC24 colorIn,
    out Euresys.Open_eVision_2_10.ERGB colorOut
)

void Dequantize (
    Euresys.Open_eVision_2_10.EC24 colorIn,
    out Euresys.Open_eVision_2_10.EXYZ colorOut
)

void Dequantize (
    Euresys.Open_eVision_2_10.EC24 colorIn,
    out Euresys.Open_eVision_2_10.EYUV colorOut
)

void Dequantize (
    Euresys.Open_eVision_2_10.EC24 colorIn,
    out Euresys.Open_eVision_2_10.EYIQ colorOut
)
```

```

void Dequantize(
    Euresys.Open_eVision_2_10.EC24 colorIn,
    out Euresys.Open_eVision_2_10.ELSH colorOut
)

void Dequantize(
    Euresys.Open_eVision_2_10.EC24 colorIn,
    out Euresys.Open_eVision_2_10.EVSH colorOut
)

void Dequantize(
    Euresys.Open_eVision_2_10.EC24 colorIn,
    out Euresys.Open_eVision_2_10.EISH colorOut
)

void Dequantize(
    Euresys.Open_eVision_2_10.EC24 colorIn,
    out Euresys.Open_eVision_2_10.EYSH colorOut
)

void Dequantize(
    Euresys.Open_eVision_2_10.EC24 colorIn,
    out Euresys.Open_eVision_2_10.ELAB colorOut
)

void Dequantize(
    Euresys.Open_eVision_2_10.EC24 colorIn,
    out Euresys.Open_eVision_2_10.ELCH colorOut
)

void Dequantize(
    Euresys.Open_eVision_2_10.EC24 colorIn,
    out Euresys.Open_eVision_2_10.ELUV colorOut
)

```

Parameters

colorIn

Input quantized color.

colorOut

Output unquantized color, as defined by the corresponding structure.

Remarks

Quantization is the process that transforms a continuous value, usually represented as a floating-point value in the **[0..1]** interval, into a discrete one, usually represented as an integer in the **[0..255]** interval.

Dequantization is the reverse process.

EasyColor.DstQuantization

Quantization mode for output values.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
static Euresys.Open_eVision_2_10.EColorQuantization DstQuantization  
  
    { get; set; }
```

Remarks

These settings remain permanent and influence the relevant quantized and unquantized color conversions (during lookup table initialization or image color transformation).

EasyColor.Format422To444

Converts a YUV 4:2:2 image to a YUV 4:4:4 image using interpolation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void Format422To444(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,  
    bool yFirst  
)
```

Parameters

sourceImage

Pointer to the input image/ROI, stored using the 16 bits per pixel format.

destinationImage

Pointer to the output image/ROI.

yFirst

Flag indicating if the format is YUYVYUYV (**TRUE**) or UYVYUYVY (**FALSE**).

Remarks

In the YUV system, it has been established that sub-sampling the chroma components does not degrade the visual image quality. The 4:4:4 format uses three bytes of information by pixel. The 4:2:2 format is such that only the U and V chroma of the even pixels are kept and they are stored with the even and odd luma, as follows: Thus, only two bytes per pixel are required. $Y_{[even]} U_{[even]} Y_{[odd]} V_{[even]}$

EasyColor.Format444To422

Converts a YUV 4:4:4 image to a YUV 4:2:2 image using filtering

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Format444To422(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    bool yFirst
)
```

Parameters

sourceImage

Pointer to the input image/ROI.

destinationImage

Pointer to the output image/ROI, stored using the 16 bits per pixel format.

yFirst

Flag indicating if the format is YUYVYUYV (**TRUE**) or UYVYUYVY (**FALSE**).

Remarks

In the YUV system, it has been established that sub-sampling the chroma components does not degrade the visual image quality. The 4:4:4 format uses three bytes of information by pixel. The 4:2:2 format is such that only the U and V chroma of the even pixels are kept and they are stored with the even and odd luma, as follows: Thus, only two bytes per pixel are required. $Y_{[even]} U_{[even]} Y_{[odd]} V_{[even]}$

EasyColor.GetComponent

Extracts one color plane, considered as a gray-level image, from a color image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetComponent(
    Euresys.Open_eVision_2_10.EROIC24 colorSourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 bWDestinationImage,
    uint component,
    Euresys.Open_eVision_2_10.EColorLookup lookup
)

void GetComponent(
    Euresys.Open_eVision_2_10.EROIC48 colorSourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 bWDestinationImage,
    uint component
)
```

Parameters

colorSourceImage

Pointer to the input color image/ROI.

bWDestinationImage

Pointers to the output gray-level component image/ROI.

component

Color component index (**0**, **1**, or **2**).

lookup

Pointer to the color lookup table, or **NULL**.

EasyColor.ImproveClassCenters

Redefines the class centers by computing the average color value of the pixels assigned to each class in the source image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ImproveClassCenters(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EC24Vector classCenters
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

classCenters

Pointer to the vector of the class centers.

Remarks

This implements a step of the K-means method for unsupervised clustering.

Color image segmentation allows you to decompose a color image in different regions by assigning a "class" (integer index) to every pixel. The nearest neighbor method is used, i.e. for each class a representative center is specified, and a given pixel is associated to the class with the closest center.

Color image segmentation can be used in conjunction with EasyObject to perform blob analysis on the segmented regions.

EasyColor.IshToRgb

Convert a color from any system to RGB.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void IshToRgb(
    Euresys.Open_eVision_2_10.EISH colorIn,
    out Euresys.Open_eVision_2_10.ERGB colorOut
)
```

```
void IshToRgb(  
    Euresys.Open_eVision_2_10.EC24 colorIn,  
    out Euresys.Open_eVision_2_10.EC24 colorOut  
)
```

Parameters

colorIn

Input color.

colorOut

Output color.

Remarks

These functions transform the color components (of a pixel) expressed in some color system to the corresponding components in another system.

EasyColor.LabToRgb

Convert a color from any system to RGB.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void LabToRgb(  
    Euresys.Open_eVision_2_10.ELAB colorIn,  
    out Euresys.Open_eVision_2_10.ERGB colorOut  
)  
  
void LabToRgb(  
    Euresys.Open_eVision_2_10.EC24 colorIn,  
    out Euresys.Open_eVision_2_10.EC24 colorOut  
)
```

Parameters

colorIn

Input color.

colorOut

Output color.

Remarks

These functions transform the color components (of a pixel) expressed in some color system to the corresponding components in another system.

EasyColor.LabToXyz

Convert a color from one system to another, including the xyz variant (reduced XYZ).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void LabToXyz (
    Euresys.Open_eVision_2_10.ELAB colorIn,
    out Euresys.Open_eVision_2_10.EXYZ colorOut
)

void LabToXyz (
    Euresys.Open_eVision_2_10.EC24 colorIn,
    out Euresys.Open_eVision_2_10.EC24 colorOut
)
```

Parameters

colorIn

Input color.

colorOut

Output color.

Remarks

These functions transform the color components (of a pixel) expressed in some color system to the corresponding components in another system.

EasyColor.LchToRgb

Convert a color from any system to RGB.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void LchToRgb (
    Euresys.Open_eVision_2_10.ELCH colorIn,
    out Euresys.Open_eVision_2_10.ERGB colorOut
)

void LchToRgb (
    Euresys.Open_eVision_2_10.EC24 colorIn,
    out Euresys.Open_eVision_2_10.EC24 colorOut
)
```

Parameters

colorIn

Input color.

colorOut

Output color.

Remarks

These functions transform the color components (of a pixel) expressed in some color system to the corresponding components in another system.

EasyColor.LshToRgb

Convert a color from any system to RGB.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void LshToRgb (
    Euresys.Open_eVision_2_10.ELSH colorIn,
    out Euresys.Open_eVision_2_10.ERGB colorOut
)
```

```
void LshToRgb (
    Euresys.Open_eVision_2_10.EC24 colorIn,
    out Euresys.Open_eVision_2_10.EC24 colorOut
)
```

Parameters

colorIn

Input color.

colorOut

Output color.

Remarks

These functions transform the color components (of a pixel) expressed in some color system to the corresponding components in another system.

EasyColor.LuvToRgb

Convert a color from any system to RGB.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void LuvToRgb (
    Euresys.Open_eVision_2_10.ELUV colorIn,
    out Euresys.Open_eVision_2_10.ERGB colorOut
)

void LuvToRgb (
    Euresys.Open_eVision_2_10.EC24 colorIn,
    out Euresys.Open_eVision_2_10.EC24 colorOut
)
```

Parameters

colorIn

Input color.

colorOut

Output color.

Remarks

These functions transform the color components (of a pixel) expressed in some color system to the corresponding components in another system.

EasyColor.LuvToXyz

Convert a color from one system to another, including the xyz variant (reduced XYZ).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void LuvToXyz (
    Euresys.Open_eVision_2_10.ELUV colorIn,
    out Euresys.Open_eVision_2_10.EXYZ colorOut
)

void LuvToXyz (
    Euresys.Open_eVision_2_10.EC24 colorIn,
    out Euresys.Open_eVision_2_10.EC24 colorOut
)
```

Parameters

colorIn

Input color.

colorOut

Output color.

Remarks

These functions transform the color components (of a pixel) expressed in some color system to the corresponding components in another system.

EasyColor.NtscGamma

NTSC gamma exponent

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
static float NtscGamma  
  
    { get; }
```

EasyColor.PalGamma

PAL gamma exponent

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
static float PalGamma  
  
    { get; }
```

EasyColor.PseudoColor

Applies the mapping defined by the pseudo-color lookup table to transform a gray-level image into a color image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```



```
void PseudoColor(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,  
    Euresys.Open_eVision_2_10.EPseudoColorLookup lookup  
)
```

Parameters

sourceImage

Pointer to the source gray-level image.

destinationImage

Pointer to the destination color image.

lookup

Pointer to the pseudo-color lookup table.

Remarks

Pseudo-coloring is a convenient way to display gray-level images with enhanced contrast: a shade of colors is associated to the shade of gray-level values. A simple way to define the shade of colors is to specify a path in color space.

In order to use pseudo-coloring, a special lookup table is used: [EPseudoColorLookup](#). It handles the mapping between the gray-level and color values.

EasyColor.Quantize

Convert an unquantized color of a given color system to a quantized color.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void Quantize(  
    Euresys.Open_eVision_2_10.ERGB colorIn,  
    out Euresys.Open_eVision_2_10.EC24 colorOut  
)
```

```
void Quantize(  
    Euresys.Open_eVision_2_10.EXYZ colorIn,  
    out Euresys.Open_eVision_2_10.EC24 colorOut  
)  
  
void Quantize(  
    Euresys.Open_eVision_2_10.EYUV colorIn,  
    out Euresys.Open_eVision_2_10.EC24 colorOut  
)  
  
void Quantize(  
    Euresys.Open_eVision_2_10.EYIQ colorIn,  
    out Euresys.Open_eVision_2_10.EC24 colorOut  
)  
  
void Quantize(  
    Euresys.Open_eVision_2_10.ELSH colorIn,  
    out Euresys.Open_eVision_2_10.EC24 colorOut  
)  
  
void Quantize(  
    Euresys.Open_eVision_2_10.EVSH colorIn,  
    out Euresys.Open_eVision_2_10.EC24 colorOut  
)  
  
void Quantize(  
    Euresys.Open_eVision_2_10.EISH colorIn,  
    out Euresys.Open_eVision_2_10.EC24 colorOut  
)  
  
void Quantize(  
    Euresys.Open_eVision_2_10.EYSH colorIn,  
    out Euresys.Open_eVision_2_10.EC24 colorOut  
)  
  
void Quantize(  
    Euresys.Open_eVision_2_10.ELAB colorIn,  
    out Euresys.Open_eVision_2_10.EC24 colorOut  
)  
  
void Quantize(  
    Euresys.Open_eVision_2_10.ELCH colorIn,  
    out Euresys.Open_eVision_2_10.EC24 colorOut  
)
```

```
void Quantize(  
    Euresys.Open_eVision_2_10.ELUV colorIn,  
    out Euresys.Open_eVision_2_10.EC24 colorOut  
)
```

Parameters

colorIn

Input unquantized color, as defined by the corresponding structure.

colorOut

Output quantized color.

Remarks

Quantization is the process that transforms a continuous value, usually represented as a floating-point value in the **[0..1]** interval, into a discrete one, usually represented as an integer in the **[0..255]** interval.

Dequantization is the reverse process.

EasyColor.RegisterPlanes

Sets a color plane of a color image by using a gray-level image as component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void RegisterPlanes(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,  
    int rShiftX,  
    int gShiftX,  
    int bShiftX,  
    int rShiftY,  
    int gShiftY,  
    int bShiftY  
)
```

Parameters

sourceImage

Pointers to the input image/ROI.

destinationImage

Pointer to the output image/ROI.

rShiftX

Horizontal shifting of the first plane (the red one in case of an RGB image), expressed in pixels.

gShiftX

Horizontal shifting of the second plane (the green one in case of an RGB image), expressed in pixels.

bShiftX

Horizontal shifting of the third plane (the blue one in case of an RGB image), expressed in pixels.

rShiftY

Vertical shifting of the first plane (the red one in case of an RGB image), expressed in pixels.

gShiftY

Vertical shifting of the second plane (the green one in case of an RGB image), expressed in pixels.

bShiftY

Vertical shifting of the third plane (the blue one in case of an RGB image), expressed in pixels.

Remarks

A color image can be seen as a set of three color planes, each corresponding to a color component. The color planes are themselves continuous tone images. An [EImageC24](#) contains three [EImageBW8](#).

EasyColor.RgbStandard

RGB definition to be used when converting between RGB and other color systems.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
static Euresys.Open_eVision_2_10.ERgbStandard RgbStandard
{ get; set; }
```

Remarks

Some variant of the color systems can be used. The [EasyColor::SrcQuantization](#) and [EasyColor::DstQuantization](#) functions are used to activate them.

These settings remain permanent and influence the relevant quantized and unquantized color conversions (during lookup table initialization or image color transformation).

EasyColor.RgbToIsh

Convert a color from RGB to another system.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void RgbToIsh(
    Euresys.Open_eVision_2_10.ERGB colorIn,
    out Euresys.Open_eVision_2_10.EISH colorOut
)

void RgbToIsh(
    Euresys.Open_eVision_2_10.EC24 colorIn,
    out Euresys.Open_eVision_2_10.EC24 colorOut
)
```

Parameters

colorIn

Input color.

colorOut

Output color, as defined by the corresponding structure.

Remarks

These functions transform the color components (of a pixel) expressed in some color system to the corresponding components in another system.

EasyColor.RgbToLab

Convert a color from RGB to another system.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void RgbToLab(
    Euresys.Open_eVision_2_10.ERGB colorIn,
    out Euresys.Open_eVision_2_10.ELAB colorOut
)

void RgbToLab(
    Euresys.Open_eVision_2_10.EC24 colorIn,
    out Euresys.Open_eVision_2_10.EC24 colorOut
)
```

Parameters

colorIn

Input color.

colorOut

Output color, as defined by the corresponding structure.

Remarks

These functions transform the color components (of a pixel) expressed in some color system to the corresponding components in another system.

EasyColor.RgbToLch

Convert a color from RGB to another system.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void RgbToLch(
    Euresys.Open_eVision_2_10.ERGB colorIn,
    out Euresys.Open_eVision_2_10.ELCH colorOut
)
```

```
void RgbToLch(  
    Euresys.Open_eVision_2_10.EC24 colorIn,  
    out Euresys.Open_eVision_2_10.EC24 colorOut  
)
```

Parameters

colorIn

Input color.

colorOut

Output color, as defined by the corresponding structure.

Remarks

These functions transform the color components (of a pixel) expressed in some color system to the corresponding components in another system.

EasyColor.RgbToLsh

Convert a color from RGB to another system.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void RgbToLsh(  
    Euresys.Open_eVision_2_10.ERGB colorIn,  
    out Euresys.Open_eVision_2_10.ELSH colorOut  
)  
  
void RgbToLsh(  
    Euresys.Open_eVision_2_10.EC24 colorIn,  
    out Euresys.Open_eVision_2_10.EC24 colorOut  
)
```

Parameters

colorIn

Input color.

colorOut

Output color, as defined by the corresponding structure.

Remarks

These functions transform the color components (of a pixel) expressed in some color system to the corresponding components in another system.

EasyColor.RgbToLuv

Convert a color from RGB to another system.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void RgbToLuv (
    Euresys.Open_eVision_2_10.ERGB colorIn,
    out Euresys.Open_eVision_2_10.ELUV colorOut
)

void RgbToLuv (
    Euresys.Open_eVision_2_10.EC24 colorIn,
    out Euresys.Open_eVision_2_10.EC24 colorOut
)
```

Parameters

colorIn

Input color.

colorOut

Output color, as defined by the corresponding structure.

Remarks

These functions transform the color components (of a pixel) expressed in some color system to the corresponding components in another system.

EasyColor.RgbToReducedXyz

Convert a color from one system to another, including the xyz variant (reduced XYZ).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void RgbToReducedXyz (
    Euresys.Open_eVision_2_10.ERGB colorIn,
    out Euresys.Open_eVision_2_10.EXYZ colorOut
)

void RgbToReducedXyz (
    Euresys.Open_eVision_2_10.EC24 colorIn,
    out Euresys.Open_eVision_2_10.EC24 colorOut
)
```

Parameters

colorIn

Input color.

colorOut

Output color.

Remarks

These functions transform the color components (of a pixel) expressed in some color system to the corresponding components in another system.

EasyColor.RgbToVsh

Convert a color from RGB to another system.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void RgbToVsh (
    Euresys.Open_eVision_2_10.ERGB colorIn,
    out Euresys.Open_eVision_2_10.EVSH colorOut
)
```

```
void RgbToVsh(  
    Euresys.Open_eVision_2_10.EC24 colorIn,  
    out Euresys.Open_eVision_2_10.EC24 colorOut  
)
```

Parameters

colorIn

Input color.

colorOut

Output color, as defined by the corresponding structure.

Remarks

These functions transform the color components (of a pixel) expressed in some color system to the corresponding components in another system.

EasyColor.RgbToXyz

Convert a color from RGB to another system.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void RgbToXyz(  
    Euresys.Open_eVision_2_10.ERGB colorIn,  
    out Euresys.Open_eVision_2_10.EXYZ colorOut  
)  
  
void RgbToXyz(  
    Euresys.Open_eVision_2_10.EC24 colorIn,  
    out Euresys.Open_eVision_2_10.EC24 colorOut  
)
```

Parameters

colorIn

Input color.

colorOut

Output color, as defined by the corresponding structure.

Remarks

These functions transform the color components (of a pixel) expressed in some color system to the corresponding components in another system.

EasyColor.RgbToYiq

Convert a color from RGB to another system.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void RgbToYiq(
    Euresys.Open_eVision_2_10.ERGB colorIn,
    out Euresys.Open_eVision_2_10.EYIQ colorOut
)

void RgbToYiq(
    Euresys.Open_eVision_2_10.EC24 colorIn,
    out Euresys.Open_eVision_2_10.EC24 colorOut
)
```

Parameters

colorIn

Input color.

colorOut

Output color, as defined by the corresponding structure.

Remarks

These functions transform the color components (of a pixel) expressed in some color system to the corresponding components in another system.

EasyColor.RgbToYsh

Convert a color from RGB to another system.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void RgbToYsh(
    Euresys.Open_eVision_2_10.ERGB colorIn,
    out Euresys.Open_eVision_2_10.EYSH colorOut
)

void RgbToYsh(
    Euresys.Open_eVision_2_10.EC24 colorIn,
    out Euresys.Open_eVision_2_10.EC24 colorOut
)
```

Parameters

colorIn

Input color.

colorOut

Output color, as defined by the corresponding structure.

Remarks

These functions transform the color components (of a pixel) expressed in some color system to the corresponding components in another system.

EasyColor.RgbToYuv

Convert a color from RGB to another system.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void RgbToYuv(
    Euresys.Open_eVision_2_10.ERGB colorIn,
    out Euresys.Open_eVision_2_10.EYUV colorOut
)
```

```
void RgbToYuv(  
    Euresys.Open_eVision_2_10.EC24 colorIn,  
    out Euresys.Open_eVision_2_10.EC24 colorOut  
)
```

Parameters

colorIn

Input color.

colorOut

Output color, as defined by the corresponding structure.

Remarks

These functions transform the color components (of a pixel) expressed in some color system to the corresponding components in another system.

EasyColor.SetComponent

Sets a color plane of a color image by using a gray-level image as component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void SetComponent(  
    Euresys.Open_eVision_2_10.EROIBW8 bWSourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 colorDestinationImage,  
    uint component  
)  
  
void SetComponent(  
    Euresys.Open_eVision_2_10.EROIBW16 bWSourceImage,  
    Euresys.Open_eVision_2_10.EROIC48 colorDestinationImage,  
    uint component  
)
```

Parameters

bWSourceImage

Pointers to the input gray level component image/ROI.

colorDestinationImage

Pointer to the output color image/ROI.

component

Color component index (**0**, **1**, or **2**).

Remarks

A color image can be seen as a set of three color planes, each corresponding to a color component. The color planes are themselves continuous tone images. An [EImageC24](#) contains three [EImageBW8](#).

EasyColor.SmpteGamma

SMPTE gamma exponent

Namespace: Euresys.Open_eVision_2_10

```
[C#]
static float SmpteGamma
{ get; }
```

EasyColor.SrcQuantization

Quantization mode for input values.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
static Euresys.Open_eVision_2_10.EColorQuantization SrcQuantization
{ get; set; }
```

Remarks

These settings remain permanent and influence the relevant quantized and unquantized color conversions (during lookup table initialization or image color transformation).

EasyColor.Transform

Applies a color transformation to a specified image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Transform(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    Euresys.Open_eVision_2_10.EColorLookup lookup
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI.

lookup

Pointer to the color lookup.

Remarks

In the first case, the transformation is defined by a color lookup. See Initialization ([EColorLookup](#)).

In the two other cases, the user defines a quantized or unquantized color transformation. No intermediate color lookup table is used.

A color image can be seen as a set of three color planes, each corresponding to a color component. The color planes are themselves continuous tone images. An [EImageC24](#) contains three [EImageBW8](#).

EasyColor.TransformBayer

Converts an image, using the transformation defined by a color lookup.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void TransformBayer(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    Euresys.Open_eVision_2_10.EColorLookup lookup,
    bool evenCol,
    bool evenRow
)
```

Parameters

sourceImage

Pointer to the source image/ROI. This image must be encoded using the Bayer color pattern.

destinationImage

Pointer to the destination image/ROI. This image must be encoded using the Bayer color pattern.

lookup

Pointer to the color lookup table holding the color adjustment transform. The lookup table must be previously set up by [EColorLookup::WhiteBalance](#) method (no other transforms are supported).

evenCol

TRUE if the leftmost destination image column can't contain blue pixels.

evenRow

TRUE if the topmost destination image row can't contain red pixels.

Remarks

By contrast with [EasyColor::Transform](#), the transformation is applied directly to Bayer-encoded data. This allows efficient processing to take place before conversion to the **C24** format.

EasyColor.VshToRgb

Convert a color from any system to RGB.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void VshToRgb (
    Euresys.Open_eVision_2_10.EVSH colorIn,
    out Euresys.Open_eVision_2_10.ERGB colorOut
)

void VshToRgb (
    Euresys.Open_eVision_2_10.EC24 colorIn,
    out Euresys.Open_eVision_2_10.EC24 colorOut
)
```

Parameters

colorIn

Input color.

colorOut

Output color.

Remarks

These functions transform the color components (of a pixel) expressed in some color system to the corresponding components in another system.

EasyColor.XyzToLab

Convert a color from one system to another, including the xyz variant (reduced XYZ).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void XyzToLab(
    Euresys.Open_eVision_2_10.EXYZ colorIn,
    out Euresys.Open_eVision_2_10.ELAB colorOut
)

void XyzToLab(
    Euresys.Open_eVision_2_10.EC24 colorIn,
    out Euresys.Open_eVision_2_10.EC24 colorOut
)
```

Parameters

colorIn

Input color.

colorOut

Output color.

Remarks

These functions transform the color components (of a pixel) expressed in some color system to the corresponding components in another system.

EasyColor.XyzToLuv

Convert a color from one system to another, including the xyz variant (reduced XYZ).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void XyzToLuv(
    Euresys.Open_eVision_2_10.EXYZ colorIn,
    out Euresys.Open_eVision_2_10.ELUV colorOut
)
```

```
void XyzToLuv(  
    Euresys.Open_eVision_2_10.EC24 colorIn,  
    out Euresys.Open_eVision_2_10.EC24 colorOut  
)
```

Parameters

colorIn

Input color.

colorOut

Output color.

Remarks

These functions transform the color components (of a pixel) expressed in some color system to the corresponding components in another system.

EasyColor.XyzToRgb

Convert a color from any system to RGB.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void XyzToRgb(  
    Euresys.Open_eVision_2_10.EXYZ colorIn,  
    out Euresys.Open_eVision_2_10.ERGB colorOut  
)  
  
void XyzToRgb(  
    Euresys.Open_eVision_2_10.EC24 colorIn,  
    out Euresys.Open_eVision_2_10.EC24 colorOut  
)
```

Parameters

colorIn

Input color.

colorOut

Output color.

Remarks

These functions transform the color components (of a pixel) expressed in some color system to the corresponding components in another system.

EasyColor.YiqToRgb

Convert a color from any system to RGB.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void YiqToRgb(
    Euresys.Open_eVision_2_10.EYIQ colorIn,
    out Euresys.Open_eVision_2_10.ERGB colorOut
)

void YiqToRgb(
    Euresys.Open_eVision_2_10.EC24 colorIn,
    out Euresys.Open_eVision_2_10.EC24 colorOut
)
```

Parameters

colorIn

Input color.

colorOut

Output color.

Remarks

These functions transform the color components (of a pixel) expressed in some color system to the corresponding components in another system.

EasyColor.YshToRgb

Convert a color from any system to RGB.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void YshToRgb (
    Euresys.Open_eVision_2_10.EYSH colorIn,
    out Euresys.Open_eVision_2_10.ERGB colorOut
)

void YshToRgb (
    Euresys.Open_eVision_2_10.EC24 colorIn,
    out Euresys.Open_eVision_2_10.EC24 colorOut
)
```

Parameters

colorIn

Input color.

colorOut

Output color.

Remarks

These functions transform the color components (of a pixel) expressed in some color system to the corresponding components in another system.

EasyColor.YuvToRgb

Convert a color from any system to RGB.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void YuvToRgb (
    Euresys.Open_eVision_2_10.EYUV colorIn,
    out Euresys.Open_eVision_2_10.ERGB colorOut
)
```

```
void YuvToRgb (
    Euresys.Open_eVision_2_10.EC24 colorIn,
    out Euresys.Open_eVision_2_10.EC24 colorOut
)
```

Parameters

colorIn

Input color.

colorOut

Output color.

Remarks

These functions transform the color components (of a pixel) expressed in some color system to the corresponding components in another system.

4.14. EasyImage Class

This class contains static properties and methods specific to the EasyImage library.

Namespace: Euresys.Open_eVision_2_10

Properties

OverlayColor

M Gets/Sets the color of the overlay in the destination image when a **BW8** Image is used as overlay source image in functions.

e

Methods

AdaptiveThreshold

Performs a locally adaptive threshold on the source image.

AlphaBlend

Draws an image over an other.

AnalyseHistogram

Returns a floating-point statistical parameter extracted from a range of gray levels in an image histogram (most/least frequent value/frequency, min/max value, count, average, standard deviation).

AnalyseHistogramBW16

Returns a floating-point statistical parameter extracted from a range of gray levels in an image histogram (most/least frequent value/frequency, min/max value, count, average, standard deviation).

Area

Counts the pixels whose values are above (or on) a threshold.

AreaDoubleThreshold

Counts the pixels whose values are comprised between (or on) two thresholds.

ArgumentImage

Prepares a lookup-table image for use for gradient argument computation.

AutoThreshold

Returns a suitable threshold value for a gray-level image binarization.

BiLevelBlackTopHatBox

Performs a top-hat filtering on a bilevel image (closed image minus source image) on a rectangular kernel.

BiLevelBlackTopHatDisk

Performs a top-hat filtering on a bilevel image (closed image minus source image) on a quasi-circular kernel.

BiLevelCloseBox

Performs a closing on a bilevel image (dilation followed by erosion) on a rectangular kernel.

BiLevelCloseDisk

Performs a closing on a bilevel image (dilation followed by erosion) on a quasi-circular kernel.

BiLevelDilateBox

Performs a dilation on a bilevel image (maximum of the pixel values in a defined neighborhood) on a rectangular kernel. For bilevel images, this maximum can either be 0 (if all pixels are black in the given neighborhood), or the maximum possible pixel value.

BiLevelDilateDisk

Performs a dilation on a bilevel image (maximum of the pixel values in a defined neighborhood) on a quasi-circular kernel. For bilevel images, this maximum can either be 0 (if all pixels are black in the given neighborhood), or the maximum possible pixel value.

BiLevelErodeBox

Performs an erosion on a bilevel image (minimum of the pixel values in a defined neighborhood) on a rectangular kernel. For bilevel images, this minimum can either be 0, or the maximum possible pixel value (if all pixels are white in the given neighborhood)

BiLevelErodeDisk

Performs an erosion on a bilevel image (minimum of the pixel values in a defined neighborhood) on a quasi-circular kernel. For bilevel images, this minimum can either be 0, or the maximum possible pixel value (if all pixels are white in the given neighborhood)

BiLevelMedian

Applies a median filter to a bilevel image (median of the gray values in a 3x3 neighborhood).

BiLevelMorphoGradientBox

Computes the morphological gradient of a bilevel image using a rectangular kernel.

BiLevelMorphoGradientDisk

Computes the morphological gradient of a bilevel image using a quasi-circular kernel.

BiLevelOpenBox

Performs an opening on a bilevel image (erosion followed by dilation) on a rectangular kernel.

BiLevelOpenDisk

Performs an opening on a bilevel image (erosion followed by dilation) on a quasi-circular kernel.

BiLevelThick

Applies a thickening operation on a bilevel image, using a 3x3 kernel.

BiLevelThin

Applies a thinning operation on a bilevel image, using a 3x3 kernel.

BiLevelWhiteTopHatBox

Performs a top-hat filtering on a bilevel image (source image minus opened image) on a rectangular kernel.

BiLevelWhiteTopHatDisk

Performs a top-hat filtering on a bilevel image (source image minus opened image) on a quasi-circular kernel.

BinaryMoments

Computes the zero-th, first or second order moments on the binarized image, i.e. with a unit weight for those pixels with a value above or equal to the threshold, and zero otherwise.

BlackTopHatBox

Performs a top-hat filtering on an image (closed image minus source image) on a rectangular kernel.

BlackTopHatDisk

Performs a top-hat filtering on an image (closed image minus source image) on a quasi-circular kernel.

CloseBox

Performs a closing on an image (dilation followed by erosion) on a rectangular kernel.

CloseDisk

Performs a closing on an image (dilation followed by erosion) on a quasi-circular kernel.

Contour

Follows the *contour* of an object.

Convert

Transforms the contents of an image to an image of another type.

ConvertTo422

Turns an 8-bit gray-level image into a YUV 4:2:2 encoded color image.

ConvolGaussian

Applies Gaussian filtering (binomial weights) in rectangular kernel of odd size.

ConvolGradient

Extracts the edges of an image by summing the absolute values of the Gradient X and Gradient Y derivatives and stores the absolute value (magnitude) of the result in the destination image.

ConvolGradientX

Derives an image along X using a Gradient kernel and stores the absolute value (magnitude) of the result in the destination image.

ConvolGradientY

Derives an image along Y using a Gradient kernel and stores the absolute value (magnitude) of the result in the destination image.

ConvolHighpass1

Filters an image using a 3x3 high-pass kernel and stores the absolute value (magnitude) of the result in the destination image.

ConvolHighpass2

Filters an image using a 3x3 high-pass kernel and stores the absolute value (magnitude) of the result in the destination image.

ConvolKernel

Performs a convolution in image space, i.e. applies a convolution kernel.

ConvolLaplacian4

Takes the second derivative of an image using a 4-neighbor Laplacian kernel and stores the absolute value (magnitude) of the result in the destination image.

ConvolLaplacian8

Takes the second derivative of an image using an 8-neighbor Laplacian kernel and stores the absolute value (magnitude) of the result in the destination image.

ConvolLaplacianX

Takes the horizontal second derivative and stores the absolute value (magnitude) of the result in the destination image.

ConvLaplacianY

Takes the vertical second derivative and stores the absolute value (magnitude) of the result in the destination image.

ConvLowpass1

Filters an image using a 3x3 low-pass kernel and stores the absolute value (magnitude) of the result in the destination image.

ConvLowpass2

Filters an image using a 3x3 low-pass kernel and stores the absolute value (magnitude) of the result in the destination image.

ConvLowpass3

Filters an image using a 3x3 low-pass kernel.

ConvPrewitt

Extracts the edges of an image by summing the absolute values of the Prewitt X and Prewitt Y derivatives and stores the absolute value (magnitude) of the result in the destination image.

ConvPrewittX

Derives an image along X using a Prewitt kernel and stores the absolute value (magnitude) of the result in the destination image.

ConvPrewittY

Derives an image along Y using a Prewitt kernel and stores the absolute value (magnitude) of the result in the destination image.

ConvRoberts

The Roberts edge extraction filter is based on a 2x2 kernel.

ConvSobel

Extracts the edges of an image by summing the absolute values of the Sobel X and Sobel Y derivatives.

ConvSobelX

Derives an image along X using a Sobel kernel and stores the absolute value (magnitude) of the result in the destination image.

ConvSobelY

Derives an image along Y using a Sobel kernel and stores the absolute value (magnitude) of the result in the destination image.

ConvSymmetricKernel

Performs a convolution in image space, i.e. applies a square symmetric convolution kernel of size 3x3, 5x5 or 7x7.

ConvUniform

Applies strong low-pass filtering to an image by using a uniform rectangular kernel of odd size.

Copy

Copies a source image or a constant in a destination image.

CumulateHistogram

Cumulates histogram values in another histogram.

DilateBox

Performs a dilation on an image (maximum of the gray values in a defined neighborhood) on a rectangular kernel.

DilateDisk

Performs a dilation on an image (maximum of the gray values in a defined neighborhood) on a quasi-circular kernel.

Distance

Computes the morphological distance function on a binary image (0 for black, non 0 for white).

DoubleThreshold

Converts an image by setting all pixels below the low threshold to a low value, all pixels above the high threshold to a high value, and the remaining pixels to an intermediate value.

Equalize

Equalizes an image histogram (the gray levels are re-mapped so that the histogram becomes as close to uniform as possible).

ErodeBox

Performs an erosion on an image (minimum of the gray values in a defined neighborhood) on a rectangular kernel.

ErodeDisk

Performs an erosion on an image (minimum of the gray values in a defined neighborhood) on a quasi-circular kernel.

Focusing

Returns a measure of the focusing of an image by computing the total gradient energy.

Gain

Transforms an image, applying a gain and offset to all pixels.

GainOffset

Transforms an image, applying a gain and offset to all pixels.

GetFrame

Extracts the frame of given parity from an image.

GetProfilePeaks

Detects peaks in a gray-level profile. Maxima as well as minima are considered.

GradientScalar

Computes the (scalar) gradient image derived from a given source image.

GravityCenter

Computes the coordinates of the gravity center of an image, i.e. the average coordinates of the pixels above (or on) the threshold.

HDRFusion

Fuses two images using HDR principles.

Histogram

Computes the histogram of an image (count of each gray-level value).

HistogramThreshold

Determines an appropriate threshold level based on the histogram contents, using an automatic threshold mode.

HistogramThresholdBW16

Determines an appropriate threshold level based on the histogram contents, using an automatic threshold mode.

HitAndMiss

Apply the morphological hit-and-miss transform to detect a particular pattern of foreground and background pixels in a BW8, BW16 or C24 image/ROI.

HorizontalMirror

Mirrors an image horizontally (the columns are swapped).

ImageToLineSegment

Copies the pixel values along a given line segment (arbitrarily oriented) to a vector.

ImageToPath

Given a path described by coordinates in a path vector, copies the corresponding pixel values into the same vector.

IsodataThreshold

Computes a suitable threshold value for a histogram.

IsodataThresholdBW16

Computes a suitable threshold value for a histogram.

LinearTransform

Applies a general affine transformation.

LineSegmentToImage

Copies the pixel values from a vector or a constant to the pixels of a given line segment (arbitrarily oriented).

LocalAverage

Computes the average in a rectangular window centered on every pixel.

LocalDeviation

Computes the standard deviation in a rectangular window centered on every pixel.

Lut	Transforms the gray levels of an image, using a lookup table stored in a vector (of unsigned values).
MatchFrames	Determines the optimal shift amplitude by comparing two successive lines of the image.
Median	Applies a median filter to an image (median of the gray values in a 3x3 neighborhood).
ModulusImage	Prepares a lookup-table image for use for gradient magnitude computation.
MorphoGradientBox	Computes the morphological gradient of an image using a rectangular kernel.
MorphoGradientDisk	Computes the morphological gradient of an image using a quasi-circular kernel.
Normalize	Normalizes an image, i.e. applies a linear transform to the gray levels so that their average and standard deviation are imposed.
Offset	Transforms an image, applying a gain and offset to all pixels.

OpenBox

Performs an opening on an image (erosion followed by dilation) on a rectangular kernel.

OpenDisk

Performs an opening on an image (erosion followed by dilation) on a quasi-circular kernel.

Oper

Applies the desired arithmetic or logic pixel-wise operator between two images or constants.

Overlay

Overlays an image on the top of a color image, at a given position.

PathToImage

Given a path described by coordinates in a path vector, copies the pixel values from the path vector to the corresponding image pixels.

PixelAverage

Computes the average pixel value in a gray-level or color image.

PixelCompare

Counts the number of pixels differing between two images.

PixelCount

Counts the pixels in the three value classes separated by two thresholds.

PixelMax

Computes the maximum gray-level value in an image.

PixelMaxBW16

Computes the maximum gray-level value in an image.

PixelMaxBW8

Computes the maximum gray-level value in an image.

PixelMin

Computes the minimum gray-level value in an image.

PixelMinBW16

Computes the minimum gray-level value in an image.

PixelMinBW8

Computes the minimum gray-level value in an image.

PixelStat

Computes the minimum, maximum and average gray-level values in an image.

PixelStatBW16

Computes the minimum, maximum and average gray-level values in an image.

PixelStatBW8

Computes the minimum, maximum and average gray-level values in an image.

PixelStdDev

Computes the average gray-level or color value in an image, the standard deviation of the color components, and the correlation between the color components (in the case of color images).

PixelVariance

For a gray-level image, computes the mean and variance of the pixel values.

ProfileDerivative

Computes the first derivative of a profile extracted from a gray-level image.

ProjectOnAColumn	Projects an image horizontally onto a column.
ProjectOnARow	Projects an image vertically onto a row.
RealignFrame	Shifts one frame of the image horizontally.
RebuildFrame	Rebuilds one frame of the image by interpolation between the lines of the other frame.
RecursiveAverage	Applies stronger noise reduction to small variations and conversely.
Register	Registers an image by realigning one, two or three pivot points to reference positions.
RmsNoise	Computes the root-mean-square amplitude of noise, by comparing a given image to a reference image.
ScaleRotate	Re-scales an image by an arbitrary factor and/or rotates it by an arbitrary angle.
SetCircleWarp	Prepares suitable warp images for use with function EasyImage::Warp to unwarp a circular ring-wedge shape into a straight rectangle.
SetFrame	Replaces the frame of given parity in an image.

SetRecursiveAverageLUT

Pre-compute the required non-linear transfer function for noise reduction by recursive temporal averaging.

SetupEqualize

Prepares a lookup-table for image equalization, using an image histogram.

Shrink

Resizes an image to a smaller size. Pre-filtering is applied to avoid aliasing.

SignalNoiseRatio

Computes the signal to noise ratio, in dB, by comparing a given image to a reference image.

SwapFrames

Interchanges the even and odd rows of an image.

Thick

Applies a thickening operation on an image, using a 3x3 kernel.

Thin

Applies a thinning operation on an image, using a 3x3 kernel.

ThreeLevelsMinResidueThreshold

Computes the two threshold values used to separate the pixels of an image in three classes.

Threshold

Binarize an image by setting pixels to two different possible values in a destination image, according to their value in a source image.

TwoLevelsMinResidueThreshold

Computes the threshold value used to separate the pixels of an image in two classes.

Uniformize

Shading correction is the process of transforming the gray or color component values of an image, using one or two reference images or vectors.

VerticalMirror

Mirrors an image vertically (the rows are swapped).

Warp

Transforms an image so that each pixels are moved to the locations specified in the "warp" images used as look-up tables.

WeightedMoments

Computes the zero-th, first, second, third or fourth order weighted moments on the gray-level image. The weight of a pixel is its gray-level value.

WhiteTopHatBox

Performs a top-hat filtering on an image (source image minus opened image) on a rectangular kernel.

WhiteTopHatDisk

Performs a top-hat filtering on an image (source image minus opened image) on a quasi-circular kernel.

EasyImage.AdaptiveThreshold

Performs a locally adaptive threshold on the source image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AdaptiveThreshold(
    Euresys.Open_eVision_2_10.EROIBW8 src,
    Euresys.Open_eVision_2_10.EROIBW8 dst,
    Euresys.Open_eVision_2_10.EAdaptiveThresholdMethod method,
    int halfKernelSize,
    int constant
)
```

Parameters

src

-

dst

-

method

The thresholding mode, as defined by the enumeration [EAdaptiveThresholdMethod](#).

halfKernelSize

Half width of the kernel rounded down

constant

Constant offset applied to the threshold value. By default (argument omitted) **0**, i.e. no change.

Remarks

Kernel size is always odd.

EasyImage.AlphaBlend

Draws an image over an other.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AlphaBlend(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    double opacity
)
```

Parameters

sourceImage

Foreground image.

destinationImage

Background image.

opacity

Opacity of the foreground image.

EasyImage.AnalyseHistogram

Returns a floating-point statistical parameter extracted from a range of gray levels in an image histogram (most/least frequent value/frequency, min/max value, count, average, standard deviation).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float AnalyseHistogram(
    Euresys.Open_eVision_2_10.EBWHistogramVector histogram,
    Euresys.Open_eVision_2_10.EHistogramFeature operation,
    int minimumIndex,
    int maximumIndex
)
```

Parameters

histogram

Pointer to the histogram vector.

operation

Parameter to be computed, as defined by [EHistogramFeature](#).

minimumIndex

Starting index of the gray-level range.

maximumIndex

Ending index of the gray-level range.

EasyImage.AnalyseHistogramBW16

Returns a floating-point statistical parameter extracted from a range of gray levels in an image histogram (most/least frequent value/frequency, min/max value, count, average, standard deviation).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float AnalyseHistogramBW16 (
    Euresys.Open_eVision_2_10.EBWHistogramVector histogram,
    Euresys.Open_eVision_2_10.EHistogramFeature operation,
    int minimumIndex,
    int maximumIndex
)
```

Parameters

histogram

Pointer to the histogram vector.

operation

Parameter to be computed, as defined by [EHistogramFeature](#).

minimumIndex

Starting index of the gray-level range.

maximumIndex

Ending index of the gray-level range.

EasyImage.Area

Counts the pixels whose values are above (or on) a threshold.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void Area(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EBW8 threshold,
    out int numberOfPixelsAboveThreshold
)

void Area(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EBW16 threshold,
    out int numberOfPixelsAboveThreshold
)

void Area(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    Euresys.Open_eVision_2_10.EBW8 threshold,
    out int numberOfPixelsAboveThreshold
)

void Area(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    Euresys.Open_eVision_2_10.EBW16 threshold,
    out int numberOfPixelsAboveThreshold
)

void Area(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    Euresys.Open_eVision_2_10.EBW8 threshold,
    out int numberOfPixelsAboveThreshold
)
```

```
void Area(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    Euresys.Open_eVision_2_10.EBW16 threshold,  
    out int numberOfPixelsAboveThreshold  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

threshold

The pixel thresholding value used to count the pixels

numberOfPixelsAboveThreshold

Reference to the count of pixels above or equal to the threshold.

region

Pointer to a region to apply the function only on a particular region in the image.

mask

Pointer to a mask to apply the function only on a particular region in the image. Note: the mask must have the same size as the source image.

EasyImage.AreaDoubleThreshold

Counts the pixels whose values are comprised between (or on) two thresholds.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void AreaDoubleThreshold(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EBW8 lowThreshold,  
    Euresys.Open_eVision_2_10.EBW8 highThreshold,  
    out int numberOfPixelsBetweenThresholds  
)
```

```

void AreaDoubleThreshold(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EBW16 lowThreshold,
    Euresys.Open_eVision_2_10.EBW16 highThreshold,
    out int numberOfPixelsBetweenThresholds
)

void AreaDoubleThreshold(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    Euresys.Open_eVision_2_10.EBW8 lowThreshold,
    Euresys.Open_eVision_2_10.EBW8 highThreshold,
    out int numberOfPixelsBetweenThresholds
)

void AreaDoubleThreshold(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    Euresys.Open_eVision_2_10.EBW16 lowThreshold,
    Euresys.Open_eVision_2_10.EBW16 highThreshold,
    out int numberOfPixelsBetweenThresholds
)

void AreaDoubleThreshold(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    Euresys.Open_eVision_2_10.EBW8 lowThreshold,
    Euresys.Open_eVision_2_10.EBW8 highThreshold,
    out int numberOfPixelsBetweenThresholds
)

void AreaDoubleThreshold(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    Euresys.Open_eVision_2_10.EBW16 lowThreshold,
    Euresys.Open_eVision_2_10.EBW16 highThreshold,
    out int numberOfPixelsBetweenThresholds
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

lowThreshold

Inferior threshold.

highThreshold

Superior threshold.

numberOfPixelsBetweenThresholds

Reference to the count of pixels that are above or equal to the inferior threshold, and strictly below the superior threshold.

region

Pointer to a region to apply the function only on a particular region in the image.

mask

Pointer to a mask to apply the function only on a particular region in the image. Note: the mask must have the same size as the source image.

EasyImage.ArgumentImage

Prepares a lookup-table image for use for gradient argument computation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ArgumentImage (
    Euresys.Open_eVision_2_10.EImageBW8 destinationImage,
    Euresys.Open_eVision_2_10.EBw8 phase,
    float period
)

void ArgumentImage (
    Euresys.Open_eVision_2_10.EImageBW8 destinationImage
)

void ArgumentImage (
    Euresys.Open_eVision_2_10.EImageBW8 destinationImage,
    Euresys.Open_eVision_2_10.EBw8 phase
)
```

Parameters

destinationImage

Pointer to the destination image.

phase

Argument value corresponding to the horizontal direction, in 256-th (65,536-th) of the period (by default, **phase = 0**).

period

Range of argument values corresponding to the **0..255 (0..65535)** interval, in the current angle unit (by default, **period = 0**).

Remarks

The scale and phase of the gradient argument can be adjusted. The argument angles are counted clockwise on a **0..255** scale in the **BW8** context and on a **0..65535** scale in the **BW16** one, corresponding to a specified range (full turn by default, specified period otherwise). The argument phase is counted on a **0..255** scale or on a **0..65535** scale too. Angle values outside the **0..255 (0..65535)** interval are wrapped. The period length is given in the current angle unit.

[EasyImage::ArgumentImage](#) sets a lookup-table image for use with function [EasyImage::GradientScalar](#), ready to compute the argument of the gradient in the source image, i.e. its direction. The argument will be returned as a value in range **0..255** suitable for storage in an [EImageBW8](#) or as a value in the range **0..65535** suitable for storage in an [EImageBW16](#). The phase of the argument can be adjusted.

EasyImage.AutoThreshold

Returns a suitable threshold value for a gray-level image binarization.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
Euresys.Open_eVision_2_10.EBW8 AutoThreshold(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EThresholdMode thresholdMode,  
    float relativeThresholdMode  
)  
  
Euresys.Open_eVision_2_10.EBW16 AutoThreshold(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EThresholdMode thresholdMode,  
    float relativeThresholdMode  
)  
  
Euresys.Open_eVision_2_10.EBW8 AutoThreshold(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    Euresys.Open_eVision_2_10.EThresholdMode thresholdMode,  
    float relativeThresholdMode  
)
```

```
Euresys.Open_eVision_2_10.EBW16 AutoThreshold(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    Euresys.Open_eVision_2_10.ETHresholdMode thresholdMode,  
    float relativeThresholdMode  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

thresholdMode

The thresholding mode, as defined by the enumeration [ETHresholdMode](#). To use absolute thresholding, use directly the threshold value instead.

relativeThresholdMode

Fraction of the image pixels that will be set below the threshold. Only used when the threshold value is [Relative](#) (by default, **relativeThresholdMode = 0.5**).

mask

Pointer to a mask to apply the function only on a particular region in the image. Note: the mask must have the same size as the source image.

Remarks

Several modes are available: absolute (the threshold value is given readily in the **thresholdMode** parameter), relative (the threshold value is computed to obtain a desired fraction of the image pixels) or automatic (using three different criteria).

It is possible that, in the automatic or relative thresholding modes, the computed threshold exceeds the dynamic range of the return type. In this case, the value is clipped to the maximum value that is representable in the return type.

EasyImage.BiLevelBlackTopHatBox

Performs a top-hat filtering on a bilevel image (closed image minus source image) on a rectangular kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void BiLevelBlackTopHatBox(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. Must not be the same as the source image.

halfOfKernelWidth

Half of the box width minus one (by default, **halfOfKernelWidth = 1**; **0** is allowed).

halfOfKernelHeight

Half of the box height minus one (by default, same as **halfOfKernelWidth**; **0** is allowed).

Remarks

This filter enhances the thin black features.

EasyImage.BiLevelBlackTopHatDisk

Performs a top-hat filtering on a bilevel image (closed image minus source image) on a quasi-circular kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void BiLevelBlackTopHatDisk(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth
)
```


Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. Must not be the same as the source image.

halfOfKernelWidth

Half width of the kernel minus one (by default, **halfOfKernelWidth = 1**; **0** is allowed).

Remarks

This filter enhances the thin black features.

EasyImage.BiLevelCloseBox

Performs a closing on a bilevel image (dilation followed by erosion) on a rectangular kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void BiLevelCloseBox(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

halfOfKernelWidth

Half of the box width minus one (by default, **halfOfKernelWidth = 1**; **0** is allowed).

halfOfKernelHeight

Half of the box height minus one (by default, same as **halfOfKernelWidth**; **0** is allowed).

EasyImage.BiLevelCloseDisk

Performs a closing on a bilevel image (dilation followed by erosion) on a quasi-circular kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void BiLevelCloseDisk(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

halfOfKernelWidth

Half width of the kernel minus one (by default, **halfOfKernelWidth = 1**; **0** is allowed).

EasyImage.BiLevelDilateBox

Performs a dilation on a bilevel image (maximum of the pixel values in a defined neighborhood) on a rectangular kernel.

For bilevel images, this maximum can either be 0 (if all pixels are black in the given neighborhood), or the maximum possible pixel value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void BiLevelDilateBox(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

halfOfKernelWidth

Half of the box width minus one (by default, **halfOfKernelWidth = 1**; **0** is allowed).

halfOfKernelHeight

Half of the box height minus one (by default, same as **halfOfKernelWidth**; **0** is allowed).

EasyImage.BiLevelDilateDisk

Performs a dilation on a bilevel image (maximum of the pixel values in a defined neighborhood) on a quasi-circular kernel.

For bilevel images, this maximum can either be 0 (if all pixels are black in the given neighborhood), or the maximum possible pixel value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void BiLevelDilateDisk(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

halfOfKernelWidth

Half width of the kernel minus one (by default, **halfOfKernelWidth = 1**; **0** is allowed).

EasyImage.BiLevelErodeBox

Performs an erosion on a bilevel image (minimum of the pixel values in a defined neighborhood) on a rectangular kernel.

For bilevel images, this minimum can either be 0, or the maximum possible pixel value (if all pixels are white in the given neighborhood)

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void BiLevelErodeBox(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,  
    uint halfOfKernelWidth,  
    uint halfOfKernelHeight  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

halfOfKernelWidth

Half of the box width minus one (by default, **halfOfKernelWidth = 1**; **0** is allowed).

halfOfKernelHeight

Half of the box height minus one (by default, same as **halfOfKernelWidth**; **0** is allowed).

EasyImage.BiLevelErodeDisk

Performs an erosion on a bilevel image (minimum of the pixel values in a defined neighborhood) on a quasi-circular kernel.

For bilevel images, this minimum can either be 0, or the maximum possible pixel value (if all pixels are white in the given neighborhood)

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void BiLevelErodeDisk(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

halfOfKernelWidth

Half width of the kernel minus one (by default, **halfOfKernelWidth = 1**; **0** is allowed).

EasyImage.BiLevelMedian

Applies a median filter to a bilevel image (median of the gray values in a 3x3 neighborhood).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void BiLevelMedian(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. Must not be the same as source image.

EasyImage.BiLevelMorphoGradientBox

Computes the morphological gradient of a bilevel image using a rectangular kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void BiLevelMorphoGradientBox(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

halfOfKernelWidth

Half of the box width minus one (by default, **halfOfKernelWidth = 1**; **0** is allowed).

halfOfKernelHeight

Half of the box height minus one (by default, same as **halfOfKernelWidth**; **0** is allowed).

Remarks

The morphological gradient is the difference between the dilation and the erosion of the image, using the same structuring element.

The kernel size is a pair of odd numbers; they must be halved before they are passed. For instance, a 3x5 kernel is passed as 1x2.

EasyImage.BiLevelMorphoGradientDisk

Computes the morphological gradient of a bilevel image using a quasi-circular kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void BiLevelMorphoGradientDisk(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

halfOfKernelWidth

Half width of the kernel minus one (by default, **halfOfKernelWidth = 1**; **0** is allowed).

Remarks

The morphological gradient is the difference between the dilation and the erosion of the image, using the same structuring element.

EasyImage.BiLevelOpenBox

Performs an opening on a bilevel image (erosion followed by dilation) on a rectangular kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void BiLevelOpenBox(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

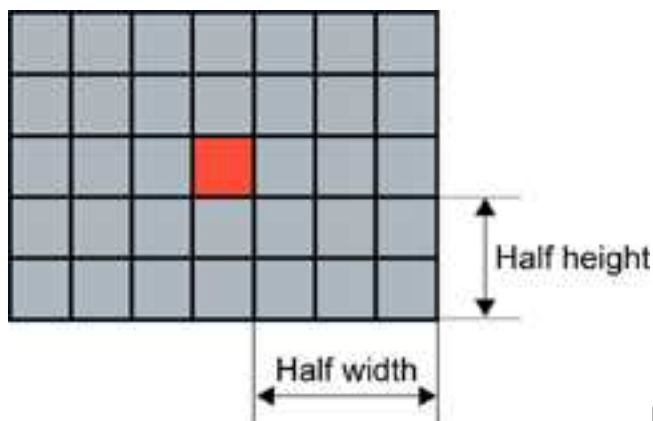
halfOfKernelWidth

Half of the box width minus one, as shown on the picture below (by default, **halfOfKernelWidth =1**; **0** is allowed).

halfOfKernelHeight

Half of the box height minus one, as shown on the picture below (by default, same as **halfOfKernelWidth**; **0** is allowed).

Remarks



Rectangular kernel of half width = 3 and half height = 2

EasyImage.BiLevelOpenDisk

Performs an opening on a bilevel image (erosion followed by dilation) on a quasi-circular kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void BiLevelOpenDisk(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

halfOfKernelWidth

Half width of the kernel minus one, as shown on the picture below (by default, **halfOfKernelWidth = 1**; **0** is allowed).

EasyImage.BiLevelThick

Applies a thickening operation on a bilevel image, using a 3x3 kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void BiLevelThick(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    Euresys.Open_eVision_2_10.EKernel thickeningKernel,
    Euresys.Open_eVision_2_10.EKernelRotation rotationMode,
    ref int numberOfIterations
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. Must not be the same as source image.

thickeningKernel

Pointer to the thickening kernel.

rotationMode

Rotation mode, as defined by [EKernelRotation](#).

numberOfIterations

Number of iterations to apply. **0** indicates stop when convergence is reached. Upon return, gives the number of passes actually performed. If the rotation mode is set to either [Clockwise](#) or [Anticlockwise](#), a pass comprises eight kernel rotations.

Remarks

The thickening kernel coefficients must be **0** (matching black pixel, value 0), **1** (matching non black pixel, value > 0) or **-1** (don't care). When a match is found between the kernel coefficients and the neighborhood of a pixel, the pixel value is set to **255**.

EasyImage.BiLevelThin

Applies a thinning operation on a bilevel image, using a 3x3 kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void BiLevelThin(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    Euresys.Open_eVision_2_10.EKernel thinningKernel,
    Euresys.Open_eVision_2_10.EKernelRotation rotationMode,
    ref int numberOfIterations
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. Must not be the same as source image.

thinningKernel

Pointer to the thinning kernel.

rotationMode

Rotation mode, as defined by [EKernelRotation](#).

numberOfIterations

Number of iterations to apply. **0** indicates stop when convergence is reached. Upon return, gives the number of passes actually performed. If the rotation mode is set to either [Clockwise](#) or [Anticlockwise](#), a pass comprises eight kernel rotations.

Remarks

The thinning kernel coefficients must be **0** (matching black pixel, value 0), **1** (matching non black pixel, value > 0) or **-1** (don't care). When a match is found between the kernel coefficients and the neighborhood of a pixel, the pixel value is set to 0.

EasyImage.BiLevelWhiteTopHatBox

Performs a top-hat filtering on a bilevel image (source image minus opened image) on a rectangular kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void BiLevelWhiteTopHatBox(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. Must not be the same as the source image.

halfOfKernelWidth

Half of the box width minus one (by default, **halfOfKernelWidth = 1**; **0** is allowed).

halfOfKernelHeight

Half of the box height minus one (by default, same as **halfOfKernelWidth**; **0** is allowed).

Remarks

This filter enhances the thin white features.

EasyImage.BiLevelWhiteTopHatDisk

Performs a top-hat filtering on a bilevel image (source image minus opened image) on a quasi-circular kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void BiLevelWhiteTopHatDisk(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. Must not be the same as the source image.

halfOfKernelWidth

Half width of the kernel minus one (by default, **halfOfKernelWidth = 1**; 0 is allowed).

Remarks

This filter enhances the thin white features.

EasyImage.BinaryMoments

Computes the zero-th, first or second order moments on the binarized image, i.e. with a unit weight for those pixels with a value above or equal to the threshold, and zero otherwise.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void BinaryMoments(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    uint threshold,
    out float M,
    out float Mx,
    out float My
)
```

```
void BinaryMoments(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    uint threshold,  
    out float M,  
    out float Mx,  
    out float My  
)
```

```
void BinaryMoments(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.ERegion region,  
    uint threshold,  
    out float M,  
    out float Mx,  
    out float My  
)
```

```
void BinaryMoments(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.ERegion region,  
    uint threshold,  
    out float M,  
    out float Mx,  
    out float My  
)
```

```
void BinaryMoments(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    uint threshold,  
    out float M,  
    out float Mx,  
    out float My  
)
```

```
void BinaryMoments(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    uint threshold,  
    out float M,  
    out float Mx,  
    out float My  
)
```

```
void BinaryMoments (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    uint threshold,
    out float M,
    out float Mx,
    out float My,
    out float Mxx,
    out float Mxy,
    out float Myy
)

void BinaryMoments (
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    uint threshold,
    out float M,
    out float Mx,
    out float My,
    out float Mxx,
    out float Mxy,
    out float Myy
)

void BinaryMoments (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImageconst,
    Euresys.Open_eVision_2_10.ERegion region,
    uint threshold,
    out float M,
    out float Mx,
    out float My,
    out float Mxx,
    out float Mxy,
    out float Myy
)

void BinaryMoments (
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    uint threshold,
    out float M,
    out float Mx,
    out float My,
    out float Mxx,
    out float Mxy,
    out float Myy
)
```

```

void BinaryMoments (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    uint threshold,
    out float M,
    out float Mx,
    out float My,
    out float Mxx,
    out float Mxy,
    out float Myy
)

void BinaryMoments (
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    uint threshold,
    out float M,
    out float Mx,
    out float My,
    out float Mxx,
    out float Mxy,
    out float Myy
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

threshold

Binarization threshold.

M

Reference to the zero-th order moment (area).

Mx

Reference to the first-order, uncentered moments (weighted sum of abscissas).

My

Reference to the first-order, uncentered moments (weighted sum of ordinates).

region

Pointer to a region to apply the function only on a particular region in the image.

mask

Pointer to a mask to apply the function only on a particular region in the image. Note: the mask must have the same size as the source image.

Mxx

Reference to the second-order, uncentered moments (weighted sum of squared abscissas).

Mxy

Reference to the second-order, uncentered moments (weighted sum of cross-product of abscissas and ordinates).

Myy

Reference to the second-order, uncentered moments (weighted sum of squared ordinates).

sourceImageconst

-

EasyImage.BlackTopHatBox

Performs a top-hat filtering on an image (closed image minus source image) on a rectangular kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void BlackTopHatBox(
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW1 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)

void BlackTopHatBox(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)

void BlackTopHatBox(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)
```

```
void BlackTopHatBox(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,  
    uint halfOfKernelWidth,  
    uint halfOfKernelHeight  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. Must not be the same as the source image.

halfOfKernelWidth

Half of the box width minus one (by default, **halfOfKernelWidth = 1**; **0** is allowed).

halfOfKernelHeight

Half of the box height minus one (by default, same as **halfOfKernelWidth**; **0** is allowed).

Remarks

This filter enhances the thin black features.

EasyImage.BlackTopHatDisk

Performs a top-hat filtering on an image (closed image minus source image) on a quasi-circular kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void BlackTopHatDisk(  
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW1 destinationImage,  
    uint halfOfKernelWidth  
)
```

```

void BlackTopHatDisk(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth
)

void BlackTopHatDisk(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    uint halfOfKernelWidth
)

void BlackTopHatDisk(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    uint halfOfKernelWidth
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. Must not be the same as the source image.

halfOfKernelWidth

Half width of the kernel minus one (by default, **halfOfKernelWidth = 1**; **0** is allowed).

Remarks

This filter enhances the thin black features.

EasyImage.CloseBox

Performs a closing on an image (dilation followed by erosion) on a rectangular kernel.

Namespace: Euresys.Open_eVision_2_10

[C#]

```

void CloseBox(
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW1 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)

void CloseBox(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)

void CloseBox(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)

void CloseBox(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

halfOfKernelWidth

Half of the box width minus one (by default, **halfOfKernelWidth = 1**; **0** is allowed).

halfOfKernelHeight

Half of the box height minus one (by default, same as **halfOfKernelWidth**; **0** is allowed).

EasyImage.CloseDisk

Performs a closing on an image (dilation followed by erosion) on a quasi-circular kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void CloseDisk(
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW1 destinationImage,
    uint halfOfKernelWidth
)

void CloseDisk(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth
)

void CloseDisk(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    uint halfOfKernelWidth
)

void CloseDisk(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    uint halfOfKernelWidth
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

halfOfKernelWidth

Half width of the kernel minus one (by default, **halfOfKernelWidth = 1**; **0** is allowed).

EasyImage.Contour

Follows the *contour* of an object.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void Contour(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EContourMode contourMode,  
    int startX,  
    int startY,  
    Euresys.Open_eVision_2_10.EContourThreshold thresholdMode,  
    uint threshold,  
    Euresys.Open_eVision_2_10.EConnexity connexity,  
    Euresys.Open_eVision_2_10.EPathVector path  
)
```

```
void Contour(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EContourMode contourMode,  
    int startX,  
    int startY,  
    Euresys.Open_eVision_2_10.EContourThreshold thresholdMode,  
    uint threshold,  
    Euresys.Open_eVision_2_10.EConnexity connexity,  
    Euresys.Open_eVision_2_10.EPathVector path  
)
```

```
void Contour(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EContourMode contourMode,  
    int startX,  
    int startY,  
    Euresys.Open_eVision_2_10.EContourThreshold thresholdMode,  
    uint threshold,  
    Euresys.Open_eVision_2_10.EConnexity connexity,  
    Euresys.Open_eVision_2_10.EBW8PathVector path,  
    bool freeman  
)
```

```
void Contour(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EContourMode contourMode,  
    int startX,  
    int startY,  
    Euresys.Open_eVision_2_10.EContourThreshold thresholdMode,  
    uint threshold,  
    Euresys.Open_eVision_2_10.EConnexity connexity,  
    Euresys.Open_eVision_2_10.EBW16PathVector path,  
    bool freeman  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

contourMode

Traversal mode, as defined by [EContourMode](#).

startX

Start point abscissa.

startY

Start point ordinate.

thresholdMode

Thresholding mode as defined by [EThresholdMode](#).

threshold

Threshold level.

connexity

Contour connexity, as defined by [EConnexity](#).

path

Pointer to the destination vector.

freeman

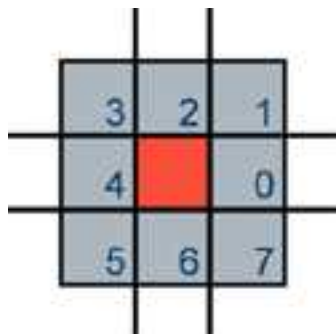
Specifies if Freeman codes are to be retrieved rather than pixel values.

Remarks

A threshold is applied so that objects become blobs. The contour is a closed or not (see property **Get/SetClosed**) connected path, forming the boundary of the blob.

When destination vector is an [EBW8PathVector](#) or a [EBW16PathVector](#), this vector can contain two different information.

If the **bFreeman** argument is **FALSE**, which is the default value, member **m_bw8(16)Pixel** in the vector elements contains the gray-level value of the contour pixels. If it is **TRUE**, the member instead gives the Freeman code leading from a pixel to next. The Freeman codes are numbered from 0 in the horizontal direction and incremented anticlockwise.



Freeman code, leading from a pixel to another adjacent pixel

EasyImage.Convert

Transforms the contents of an image to an image of another type.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Convert(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage
)

void Convert(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage
)
```



```
void Convert(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC15 destinationImage  
)  
  
void Convert(  
    Euresys.Open_eVision_2_10.EROIC15 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage  
)  
  
void Convert(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC15 destinationImage  
)  
  
void Convert(  
    Euresys.Open_eVision_2_10.EROIC15 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage  
)  
  
void Convert(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC16 destinationImage  
)  
  
void Convert(  
    Euresys.Open_eVision_2_10.EROIC16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage  
)  
  
void Convert(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC16 destinationImage  
)  
  
void Convert(  
    Euresys.Open_eVision_2_10.EROIC16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage  
)  
  
void Convert(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24A destinationImage  
)
```

```
void Convert(  
    Euresys.Open_eVision_2_10.EROIC24A sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage  
)  
  
void Convert(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW1 destinationImage  
)  
  
void Convert(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW1 destinationImage  
)  
  
void Convert(  
    Euresys.Open_eVision_2_10.EROIBW32 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW1 destinationImage  
)  
  
void Convert(  
    Euresys.Open_eVision_2_10.EROIBW32 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,  
    uint rightShift  
)  
  
void Convert(  
    Euresys.Open_eVision_2_10.EROIBW32 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,  
    uint rightShift  
)  
  
void Convert(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,  
    uint rightShift  
)  
  
void Convert(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,  
    uint leftShift  
)
```

```

void Convert(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW32 destinationImage,
    uint leftShift
)

void Convert(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW32 destinationImage,
    uint leftShift
)

void Convert(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 sourceImageAlpha,
    Euresys.Open_eVision_2_10.EROIC24A destinationImage
)

void Convert(
    Euresys.Open_eVision_2_10.EROIC24A sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImageAlpha
)

void Convert(
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    Euresys.Open_eVision_2_10.EBW8 highValue
)

void Convert(
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage
)

void Convert(
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    Euresys.Open_eVision_2_10.EBW16 highValue
)

void Convert(
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage
)

```

```

void Convert(
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW32 destinationImage,
    Euresys.Open_eVision_2_10.EBW32 highValue
)

void Convert(
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW32 destinationImage
)

void Convert(
    Euresys.Open_eVision_2_10.EROIC48 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    uint rightShift
)

void Convert(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC48 destinationImage,
    uint leftShift
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI.

rightShift

Right shift amplitude. By default, left justified data is assumed.

leftShift

Left shift amplitude. By default, left justified data is assumed.

sourceImageAlpha

Pointer to the source alpha component ([EImageBW8/EROIBW8](#)).

destinationImageAlpha

Pointer to the destination alpha component ([EImageBW8/EROIBW8](#)).

highValue

In the case of black and white source images/ROIs, indicates to which gray level the value **1** should be mapped. By default, **1** is mapped to the highest allowed value for the destination image/ROI.

Remarks

Conversion to a black and white image (BW1)

Turns an 8-bit gray-level image into a black and white image.

Turns a 16-bit gray-level image into a black and white image.

Turns a 32-bit gray-level image into a black and white image.

Source pixels whose values is 0 are converted to black. All other source pixel values are converted to white.

Conversion to a 8-bit gray-level image (BW8)

Turns a black and white image into an 8-bit gray-level image.

Turns a 16-bit gray-level image into an 8-bit gray-level image. A right shift can be applied to the 16-bit data to adjust the magnitude, depending on how the 16-bit data is organized. For instance, if the source image holds 10 significant bits right justified, a right shift of 2 is required to drop the 2 low order bits; if the source image holds 12 bits left justified, a right shift of 8 is required and the 4 low order bits will be truncated.

Turns an [EC15](#), [EC16](#) or [EC24](#) color image into an [EBW8](#) gray-level image. The 3 color components are simply averaged, giving the intensity component of the ISH color system.

Conversion to a 16-bit gray-level image (BW16)

Turns a black and white image into a 16-bit gray-level image.

Turns an 8-bit gray-level image into a 16-bit gray-level image. A left shift can be applied to the 8-bit data to adjust the magnitude, depending on how the 16-bit data is organized. For instance, if the destination image holds 10 significant bits right justified, a shift of 2 is required; if the destination image holds 12 bits left justified, a shift of 8 is required.

Conversion to a 32-bit gray-level image (BW32)

Turns a black and white image into a 32-bit gray-level image.

Conversion to color images

Turns an 8-bit gray-level image into a true color equivalent. The color components are all set equal to the corresponding gray-level value.

Converts between standard and Windows' packing RGB color formats. When converting from an [EC24](#) image to a [EC15](#) or [EC16](#) one, only the 5 (or 6) most significant bits of each color component are retained.

Converts between RGB 24-bit color image and RGB32 (also known as RGBA) color image. When converting from [EC24](#) to [EC24A](#), you can choose to provide or not the alpha component. On the other hand, when converting from [EC24A](#) to [EC24](#), you can choose to conserve or not the alpha component. The alpha component is retrieved and set using an [EImageBW8/EROIBW8](#).

EasyImage.ConvertTo422

Turns an 8-bit gray-level image into a YUV 4:2:2 encoded color image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ConvertTo422(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI.

Remarks

The Y component is set to the corresponding gray-level values, while the U and V components are set to **128** (achromatic light).

EasyImage.ConvolGaussian

Applies Gaussian filtering (binomial weights) in rectangular kernel of odd size.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ConvolGaussian(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)
```

```

void ConvolveGaussian(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)

void ConvolveGaussian(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)

void ConvolveGaussian(
    Euresys.Open_eVision_2_10.EBW8Vector sourceImage,
    Euresys.Open_eVision_2_10.EBW8Vector destinationImage,
    uint halfOfKernelWidth
)

void ConvolveGaussian(
    Euresys.Open_eVision_2_10.EBW16Vector sourceImage,
    Euresys.Open_eVision_2_10.EBW16Vector destinationImage,
    uint halfOfKernelWidth
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

halfOfKernelWidth

Half of the box width minus one (by default, **halfOfKernelWidth = 1**; **0** is allowed).

halfOfKernelHeight

Half of the box height minus one (by default, same as **halfOfKernelHeight**; **0** is allowed).

EasyImage.ConvolGradient

Extracts the edges of an image by summing the absolute values of the Gradient X and Gradient Y derivatives and stores the absolute value (magnitude) of the result in the destination image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ConvolGradient(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage
)

void ConvolGradient(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage
)

void ConvolGradient(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

EasyImage.ConvolGradientX

Derives an image along X using a Gradient kernel and stores the absolute value (magnitude) of the result in the destination image.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
void ConvolGradientX(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage
)

void ConvolGradientX(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage
)

void ConvolGradientX(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

Remarks

Filtering kernel: 0 0 0-1 0 1 0 0 0

EasyImage.ConvolGradientY

Derives an image along Y using a Gradient kernel and stores the absolute value (magnitude) of the result in the destination image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void ConvolGradientY(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage  
)  
  
void ConvolGradientY(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage  
)  
  
void ConvolGradientY(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

Remarks

Filtering kernel: 0 -1 00 0 00 1 0

EasyImage.ConvolHighpass1

Filters an image using a 3x3 high-pass kernel and stores the absolute value (magnitude) of the result in the destination image.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void ConvolveHighpass1(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage  
)  
  
void ConvolveHighpass1(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage  
)  
  
void ConvolveHighpass1(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

Remarks

Filtering kernel: 0 -1 0 -1 5 -1 0 -1 0

EasyImage.ConvolveHighpass2

Filters an image using a 3x3 high-pass kernel and stores the absolute value (magnitude) of the result in the destination image.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void ConvolHighpass2(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage  
)  
  
void ConvolHighpass2(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage  
)  
  
void ConvolHighpass2(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

Remarks

Filtering kernel: -1 -1 -1-1 9 -1-1 -1 -1

EasyImage.ConvolveKernel

Performs a convolution in image space, i.e. applies a convolution kernel.

Namespace: Euresys.Open_eVision_2_10

[C#]

```

void ConvolKernel (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    Euresys.Open_eVision_2_10.EKernel kernel
)

void ConvolKernel (
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    Euresys.Open_eVision_2_10.EKernel kernel
)

void ConvolKernel (
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    Euresys.Open_eVision_2_10.EKernel kernel
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI.

kernel

Pointer to the convolution kernel.

Remarks

Please note that **sourceImage** and **destinationImage** must be different objects.

EasyImage.ConvolLaplacian4

Takes the second derivative of an image using a 4-neighbor Laplacian kernel and stores the absolute value (magnitude) of the result in the destination image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Convollaplacian4(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage
)

void Convollaplacian4(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage
)

void Convollaplacian4(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

Remarks

Filtering kernel: 0 1 01 -4 10 1 0

EasyImage.Convollaplacian8

Takes the second derivative of an image using an 8-neighbor Laplacian kernel and stores the absolute value (magnitude) of the result in the destination image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void ConvLaplacian8(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage  
)  
  
void ConvLaplacian8(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage  
)  
  
void ConvLaplacian8(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

Remarks

Filtering kernel: 1 1 11 -8 11 1 1

EasyImage.ConvLaplacianX

Takes the horizontal second derivative and stores the absolute value (magnitude) of the result in the destination image.

Namespace: Euresys.Open_eVision_2_10

[C#]

```

void ConvLaplacianX(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage
)

void ConvLaplacianX(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage
)

void ConvLaplacianX(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

Remarks

Filtering kernel: 1 -2 1

EasyImage.ConvLaplacianY

Takes the vertical second derivative and stores the absolute value (magnitude) of the result in the destination image.

Namespace: Euresys.Open_eVision_2_10

```

[C#]
void ConvLaplacianY(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage
)

```



```
void ConvLaplacianY(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage  
)  
  
void ConvLaplacianY(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

Remarks

Filtering kernel: 1-2 1

EasyImage.ConvLowpass1

Filters an image using a 3x3 low-pass kernel and stores the absolute value (magnitude) of the result in the destination image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void ConvLowpass1(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage  
)
```

```
void ConvolveLowpass1(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage  
)  
  
void ConvolveLowpass1(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

Remarks

Filtering kernel: 1 1 11 1 11 1 1

EasyImage.ConvolveLowpass2

Filters an image using a 3x3 low-pass kernel and stores the absolute value (magnitude) of the result in the destination image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void ConvolveLowpass2(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage  
)
```

```

void ConvolveLowpass2(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage
)

void ConvolveLowpass2(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

Remarks

Filtering kernel: 1 1 11 0 11 1 1

EasyImage.ConvolveLowpass3

Filters an image using a 3x3 low-pass kernel.

Namespace: Euresys.Open_eVision_2_10

```

[C#]

void ConvolveLowpass3(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage
)

void ConvolveLowpass3(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage
)

```

```
void ConvolveLowpass3(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

Remarks

Filtering kernel: 1 2 12 4 21 2 1

EasyImage.ConvolvePrewitt

Extracts the edges of an image by summing the absolute values of the Prewitt X and Prewitt Y derivatives and stores the absolute value (magnitude) of the result in the destination image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void ConvolvePrewitt(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage  
)  
  
void ConvolvePrewitt(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage  
)  
  
void ConvolvePrewitt(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

EasyImage.ConvolvePrewittX

Derives an image along X using a Prewitt kernel and stores the absolute value (magnitude) of the result in the destination image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ConvolvePrewittX(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage
)

void ConvolvePrewittX(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage
)

void ConvolvePrewittX(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

Remarks

Filtering kernel: -1 0 1 -1 0 1 -1 0 1

EasyImage.ConvolvePrewittY

Derives an image along Y using a Prewitt kernel and stores the absolute value (magnitude) of the result in the destination image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ConvolvePrewittY(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage
)

void ConvolvePrewittY(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage
)

void ConvolvePrewittY(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

Remarks

Filtering kernel: -1 -1 -1 0 0 0 1 1 1

EasyImage.ConvolveRoberts

The Roberts edge extraction filter is based on a 2x2 kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ConvolRoberts (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage
)

void ConvolRoberts (
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage
)

void ConvolRoberts (
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

Remarks

It computes the sum of absolute differences of the pixel values in the diagonal directions.

EasyImage.ConvolSobel

Extracts the edges of an image by summing the absolute values of the Sobel X and Sobel Y derivatives.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```

void ConvSobel (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage
)

void ConvSobel (
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage
)

void ConvSobel (
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

EasyImage.ConvSobelX

Derives an image along X using a Sobel kernel and stores the absolute value (magnitude) of the result in the destination image.

Namespace: Euresys.Open_eVision_2_10

```

[C#]
void ConvSobelX(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage
)

```



```
void ConvSobelX(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage  
)  
  
void ConvSobelX(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

Remarks

Filtering kernel: -1 0 1-2 0 2-1 0 1

EasyImage.ConvSobelY

Derives an image along Y using a Sobel kernel and stores the absolute value (magnitude) of the result in the destination image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void ConvSobelY(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage  
)
```

```

void ConvSobely(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage
)

void ConvSobely(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

Remarks

Filtering kernel: -1 -2 -1 0 0 0 1 2 1

EasyImage.ConvSymmetricKernel

Performs a convolution in image space, i.e. applies a square symmetric convolution kernel of size 3x3, 5x5 or 7x7.

Namespace: Euresys.Open_eVision_2_10

```

[C#]

void ConvSymmetricKernel(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    Euresys.Open_eVision_2_10.EKernel kernel
)

```

```
void ConvolSymmetricKernel(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,  
    Euresys.Open_eVision_2_10.EKernel kernel  
)  
  
void ConvolSymmetricKernel(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,  
    Euresys.Open_eVision_2_10.EKernel kernel  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI.

kernel

Pointer to the convolution kernel.

Remarks

This function is a synonym for [EasyImage::ConvolKernel](#).

EasyImage.ConvolveUniform

Applies strong low-pass filtering to an image by using a uniform rectangular kernel of odd size.

Namespace: Euresys.Open_eVision_2_10

[C#]

```

void ConvolveUniform(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)

void ConvolveUniform(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)

void ConvolveUniform(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)

void ConvolveUniform(
    Euresys.Open_eVision_2_10.EBW8Vector sourceVector,
    Euresys.Open_eVision_2_10.EBW8Vector destinationVector,
    uint halfOfKernelWidth
)

void ConvolveUniform(
    Euresys.Open_eVision_2_10.EBW16Vector sourceVector,
    Euresys.Open_eVision_2_10.EBW16Vector destinationVector,
    uint halfOfKernelWidth
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image) and the default value for **un32HalfWidth (1)** has to be used.

halfOfKernelWidth

Half of the box width minus one (by default, **halfOfKernelWidth = 1**; **0** is allowed).

halfOfKernelHeight

Half of the box height minus one (by default, same as **halfOfKernelWidth**; **0** is allowed).

sourceVector

Pointer to the source vector.

destinationVector

Pointer to the destination vector. If **NULL** (default), this operation is destructive (i.e. applied to the source vector) and the default value for **un32HalfWidth (1)** has to be used.

Remarks

This filter replaces every pixel values by the arithmetic mean of the neighboring values in a rectangular window. To handle pixels along edges, the source pixels are replicated outwards as many times as required. A very nice feature of this function is that its running time does not depend on the kernel size!

EasyImage.Copy

Copies a source image or a constant in a destination image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Copy(
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW1 destinationImage
)

void Copy(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage
)

void Copy(
    Euresys.Open_eVision_2_10.EROIBW32 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW32 destinationImage
)

void Copy(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage
)
```

```
void Copy(  
    Euresys.Open_eVision_2_10.EROIC15 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC15 destinationImage  
    )  
  
void Copy(  
    Euresys.Open_eVision_2_10.EROIC16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC16 destinationImage  
    )  
  
void Copy(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage  
    )  
  
void Copy(  
    Euresys.Open_eVision_2_10.EROIC48 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC48 destinationImage  
    )  
  
void Copy(  
    Euresys.Open_eVision_2_10.EBW1 constant,  
    Euresys.Open_eVision_2_10.EROIBW1 destinationImage  
    )  
  
void Copy(  
    Euresys.Open_eVision_2_10.EBW16 constant,  
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage  
    )  
  
void Copy(  
    Euresys.Open_eVision_2_10.EBW32 constant,  
    Euresys.Open_eVision_2_10.EROIBW32 destinationImage  
    )  
  
void Copy(  
    Euresys.Open_eVision_2_10.EC24 constant,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage  
    )  
  
void Copy(  
    Euresys.Open_eVision_2_10.EC15 constant,  
    Euresys.Open_eVision_2_10.EROIC15 destinationImage  
    )
```

```

void Copy(
    Euresys.Open_eVision_2_10.EC16 constant,
    Euresys.Open_eVision_2_10.EROIC16 destinationImage
)

void Copy(
    Euresys.Open_eVision_2_10.EBW8 constant,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI.

constant

Gray-level or color constant.

EasyImage.CumulateHistogram

Cumulates histogram values in another histogram.

Namespace: Euresys.Open_eVision_2_10

```

[C#]

void CumulateHistogram(
    Euresys.Open_eVision_2_10.EBWHistogramVector sourceVector,
    Euresys.Open_eVision_2_10.EBWHistogramVector destinationVector
)

```

Parameters

sourceVector

Pointer to the source vector.

destinationVector

Pointer to the destination vector.

Remarks

Calling this function after [EasyImage::Histogram](#) allows you to compute the cumulative histogram of an image, i.e. the count of pixels below a given threshold value (instead of the count of pixels with a given gray value, as computed by [EasyImage::Histogram](#)).

EasyImage.DilateBox

Performs a dilation on an image (maximum of the gray values in a defined neighborhood) on a rectangular kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void DilateBox(
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW1 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)

void DilateBox(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)

void DilateBox(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)

void DilateBox(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)
```


Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

halfOfKernelWidth

Half of the box width minus one (by default, **halfOfKernelWidth = 1**; **0** is allowed).

halfOfKernelHeight

Half of the box height minus one (by default, same as **halfOfKernelWidth**; **0** is allowed).

EasyImage.DilateDisk

Performs a dilation on an image (maximum of the gray values in a defined neighborhood) on a quasi-circular kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DilateDisk(
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW1 destinationImage,
    uint halfOfKernelWidth
)

void DilateDisk(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth
)

void DilateDisk(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    uint halfOfKernelWidth
)
```

```
void DilateDisk(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,  
    uint halfOfKernelWidth  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

halfOfKernelWidth

Half width of the kernel minus one (by default, **halfOfKernelWidth =1**; **0** is allowed).

EasyImage.Distance

Computes the morphological distance function on a binary image (0 for black, non 0 for white).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void Distance(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EImageBW16 destinationImage,  
    int valueOutOfImage  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. Must not be the same as source image.

valueOutOfImage

Out-of-bounds image value. By default, this value is **0**.

Remarks

So, each pixel of the destination image will contain, at the end of the processing, the morphological distance of the corresponding pixel in the source image. The distance function at a given pixel tells how many erosion passes are required to set it to black.

EasyImage.DoubleThreshold

Converts an image by setting all pixels below the low threshold to a low value, all pixels above the high threshold to a high value, and the remaining pixels to an intermediate value.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void DoubleThreshold(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint lowThreshold,
    uint highThreshold,
    byte lowValue,
    byte middleValue,
    byte highValue
)

void DoubleThreshold(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    uint lowThreshold,
    uint highThreshold,
    Euresys.Open_eVision_2_10.EBW16 lowValue,
    Euresys.Open_eVision_2_10.EBW16 middleValue,
    Euresys.Open_eVision_2_10.EBW16 highValue
)

void DoubleThreshold(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    uint lowThreshold,
    uint highThreshold
)
```

```

void DoubleThreshold(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint lowThreshold,
    uint highThreshold,
    byte lowValue,
    byte middleValue,
    byte highValue
)

void DoubleThreshold(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    uint lowThreshold,
    uint highThreshold,
    Euresys.Open_eVision_2_10.EBW16 lowValue,
    Euresys.Open_eVision_2_10.EBW16 middleValue,
    Euresys.Open_eVision_2_10.EBW16 highValue
)

void DoubleThreshold(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    uint lowThreshold,
    uint highThreshold
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI.

lowThreshold

Low threshold value.

highThreshold

High threshold value.

lowValue

Value for pixels strictly below the low threshold.

middleValue

Value for pixels that are above or equal to the low threshold, and below the high threshold.

highValue

Value for pixels above or equal to the high threshold.

region

Pointer to a region to apply the function only on a particular region in the image.

EasyImage.Equalize

Equalizes an image histogram (the gray levels are re-mapped so that the histogram becomes as close to uniform as possible).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Equalize(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage
)

void Equalize(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI.

Remarks

This strongly enhances the contrast in dark areas.

EasyImage.ErodeBox

Performs an erosion on an image (minimum of the gray values in a defined neighborhood) on a rectangular kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void ErodeBox(
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW1 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)

void ErodeBox(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)

void ErodeBox(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)

void ErodeBox(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

halfOfKernelWidth

Half of the box width minus one (by default, **halfOfKernelWidth = 1**; **0** is allowed).

halfOfKernelHeight

Half of the box height minus one (by default, same as **halfOfKernelWidth**; **0** is allowed).

EasyImage.ErodeDisk

Performs an erosion on an image (minimum of the gray values in a defined neighborhood) on a quasi-circular kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ErodeDisk(
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW1 destinationImage,
    uint halfOfKernelWidth
)

void ErodeDisk(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth
)

void ErodeDisk(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    uint halfOfKernelWidth
)

void ErodeDisk(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    uint halfOfKernelWidth
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

halfOfKernelWidth

Half width of the kernel minus one (by default, **halfOfKernelWidth = 1; 0** is allowed).

EasyImage.Focusing

Returns a measure of the focusing of an image by computing the total gradient energy.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float Focusing(
    Euresys.Open_eVision_2_10.EROIBW8 image
)

float Focusing(
    Euresys.Open_eVision_2_10.EROIBW16 image
)

float Focusing(
    Euresys.Open_eVision_2_10.EROIC24 image
)
```

Parameters

image

Pointer to the source image/ROI.

Remarks

When this quantity is maximum for a given image, sharp focusing is achieved.

For more information, please refer to the section **Using Open eVision -> EasyImage - Computing Image Statistics -> Image Focus** in the documentation.

EasyImage.Gain

Transforms an image, applying a gain and offset to all pixels.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Gain(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    Euresys.Open_eVision_2_10.EColor Gain
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI.

Gain

Constant gain. By default (argument omitted) **1**, i.e. no change.

Remarks

The gain should remain close to **1**, and allows contrast adjustment of the image.

The offset can be positive or negative, and allows to adjust the image intensity. The resulting values are always saturated to range **[0..255]**.

For color images, the separate gain and offset values are specified as triple of values stored in a [EColor](#). The default values leave the image unchanged.

Internally, the computations are achieved through fixed-point arithmetic with 5 bits of precision for the fractional part. This can result in loss of precision with small gains.

EasyImage.GainOffset

Transforms an image, applying a gain and offset to all pixels.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void GainOffset(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    float gain,
    float offset
)

void GainOffset(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    float gain,
    float offset
)

void GainOffset(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    Euresys.Open_eVision_2_10.EColor gain,
    Euresys.Open_eVision_2_10.EColor offset
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI.

gain

Constant gain. By default (argument omitted) **1**, i.e. no change.

offset

Constant offset. By default (argument omitted) **0**, i.e. no change.

Remarks

The gain should remain close to **1**, and allows contrast adjustment of the image.

The offset can be positive or negative, and allows to adjust the image intensity. The resulting values are always saturated to range **[0..255]**.

For color images, the separate gain and offset values are specified as triple of values stored in a [EColor](#). The default values leave the image unchanged.

Internally, the computations are achieved through fixed-point arithmetic with 5 bits of precision for the fractional part. This can result in loss of precision with small gains.

EasyImage.GetFrame

Extracts the frame of given parity from an image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetFrame(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    bool odd
)

void GetFrame(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    bool odd
)

void GetFrame(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    bool odd
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. Must not be the same as the source image.

odd

Specifies which frame is extracted (the frame made up of all lines of the same parity as **odd**).

Remarks

The size of the destination image is determined as follows:

DstImage_Width = SrcImage_Width

DstImage_Height = (SrcImage_Height + 1 - odd) / 2

EasyImage.GetProfilePeaks

Detects peaks in a gray-level profile. Maxima as well as minima are considered.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetProfilePeaks (
    Euresys.Open_eVision_2_10.EBW8Vector profile,
    Euresys.Open_eVision_2_10.EPeakVector peaks,
    uint lowThreshold,
    uint highThreshold,
    uint minimumAmplitude,
    uint minimumArea
)

void GetProfilePeaks (
    Euresys.Open_eVision_2_10.EBW16Vector profile,
    Euresys.Open_eVision_2_10.EPeakVector peaks,
    uint lowThreshold,
    uint highThreshold,
    uint minimumAmplitude,
    uint minimumArea
)
```

Parameters

profile

Pointer to the source vector.

peaks

Pointer to the destination vector.

lowThreshold

Threshold used for the minimum peaks.

highThreshold

Threshold used for the maximum peaks.

minimumAmplitude

Minimum amplitude required for a peak to be kept (may be **0**).

minimumArea

Minimum area required for a peak to be kept (may be **0**).

Remarks

To eliminate false peaks due to noise, two selection criteria are used.

A peak is the portion of the signal that is above [below] a given threshold. The peak amplitude is defined to be the difference between the threshold value and the maximum [minimum] signal value. The peak area is defined to be the surface comprised between the signal curve and the horizontal line at the given threshold.

The result is stored in a peaks vector.

EasyImage.GradientScalar

Computes the (scalar) gradient image derived from a given source image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GradientScalar(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    Euresys.Open_eVision_2_10.EROIBW8 lookupTable
)

void GradientScalar(
    Euresys.Open_eVision_2_10.EROIBW32 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    Euresys.Open_eVision_2_10.EROIBW8 lookupTable
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI.

lookupTable

Pointer to the image/ROI used as a preset lookup-table. This lookup table can be generated by one of [EasyImage::ArgumentImage](#) or [EasyImage::ModulusImage](#), or be user-defined.

Remarks

The scalar value derived from the gradient depends on the preset lookup-table image.

The gradient of a gray-scale image corresponds to a vector, the components of which are the partial derivatives of the gray-level signal in the horizontal and vertical direction. A vector can be characterized by a direction and a length, corresponding to the gradient orientation, here called *argument*, and the gradient magnitude, here called *magnitude*. Function [EasyImage::GradientScalar](#) generates a gradient direction or gradient magnitude map (gray-level image) from a given gray-level image. For efficiency, a pre-computed lookup-table is used to define the desired transformation. This lookup-table is stored as a standard [EImageBW8/EImageBW16](#). Use one of [EasyImage::ArgumentImage](#) or [EasyImage::ModulusImage](#) once before calling [EasyImage::GradientScalar](#).

EasyImage.GravityCenter

Computes the coordinates of the gravity center of an image, i.e. the average coordinates of the pixels above (or on) the threshold.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GravityCenter(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    uint threshold,
    out float gravityX,
    out float gravityY
)

void GravityCenter(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    uint threshold,
    out float gravityX,
    out float gravityY
)
```

```

void GravityCenter(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    uint threshold,
    out float gravityX,
    out float gravityY
)

void GravityCenter(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    uint threshold,
    out float gravityX,
    out float gravityY
)

void GravityCenter(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    uint threshold,
    out float gravityX,
    out float gravityY
)

void GravityCenter(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    uint threshold,
    out float gravityX,
    out float gravityY
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

threshold

Threshold.

gravityX

Reference to the gravity center abscissa.

gravityY

Reference to the gravity center ordinate.

region

Pointer to a region to apply the function only on a particular region in the image.

mask

Pointer to a mask to apply the function only on a particular region in the image. Note: the mask must have the same size as the source image.

EasyImage.HDRFusion

Fuses two images using HDR principles.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void HDRFusion(
    Euresys.Open_eVision_2_10.EROIBW8 darkSrc,
    Euresys.Open_eVision_2_10.EROIBW8 lightSrc,
    int shutterSpeedFactor,
    Euresys.Open_eVision_2_10.EROIBW16 dst
)

void HDRFusion(
    Euresys.Open_eVision_2_10.EROIBW16 darkSrc,
    Euresys.Open_eVision_2_10.EROIBW16 lightSrc,
    int shutterSpeedFactor,
    Euresys.Open_eVision_2_10.EROIBW16 dst
)

void HDRFusion(
    Euresys.Open_eVision_2_10.EROIBW16 darkSrc,
    Euresys.Open_eVision_2_10.EROIBW16 lightSrc,
    int shutterSpeedFactor,
    Euresys.Open_eVision_2_10.EROIBW32 dst
)

void HDRFusion(
    Euresys.Open_eVision_2_10.EROIC24 darkSrc,
    Euresys.Open_eVision_2_10.EROIC24 lightSrc,
    int shutterSpeedFactor,
    Euresys.Open_eVision_2_10.EROIC48 dst
)
```



```
void HDRFusion(  
    Euresys.Open_eVision_2_10.EROIC48 darkSrc,  
    Euresys.Open_eVision_2_10.EROIC48 lightSrc,  
    int shutterSpeedFactor,  
    Euresys.Open_eVision_2_10.EROIC48 dst  
)
```

Parameters

darkSrc

Dark input image (high shutter speed).

lightSrc

Light input image (low shutter speed).

shutterSpeedFactor

Shutter speed factor between light and dark image.

dst

Destination image.

EasyImage.Histogram

Computes the histogram of an image (count of each gray-level value).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void Histogram(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EBWHistogramVector histogram  
)  
  
void Histogram(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.ERegion region,  
    Euresys.Open_eVision_2_10.EBWHistogramVector histogram  
)
```

```
void Histogram(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EBWHistogramVector histogram,  
    uint mostSignificantBit,  
    uint numberOfSignificantBits,  
    bool saturate  
)  
  
void Histogram(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.ERegion region,  
    Euresys.Open_eVision_2_10.EBWHistogramVector histogram,  
    uint mostSignificantBit,  
    uint numberOfSignificantBits,  
    bool saturate  
)  
  
void Histogram(  
    Euresys.Open_eVision_2_10.EROIBW32 sourceImage,  
    Euresys.Open_eVision_2_10.EBWHistogramVector histogram,  
    uint mostSignificantBit,  
    uint numberOfSignificantBits,  
    bool saturate  
)  
  
void Histogram(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    Euresys.Open_eVision_2_10.EBWHistogramVector histogram  
)  
  
void Histogram(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    Euresys.Open_eVision_2_10.EBWHistogramVector histogram,  
    uint mostSignificantBit,  
    uint numberOfSignificantBits,  
    bool saturate  
)
```

```
void Histogram(  
    Euresys.Open_eVision_2_10.EROIBW32 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    Euresys.Open_eVision_2_10.EBWHistogramVector histogram,  
    uint mostSignificantBit,  
    uint numberOfSignificantBits,  
    bool saturate  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

histogram

Pointer to the destination vector.

region

Pointer to a region to apply the function only on a particular region in the image.

mostSignificantBit

Index of the most significant bit of the pixels (**0** has weight 1).

numberOfSignificantBits

Number of significant bits; the histogram will possess $2^{\text{numberOfSignificantBits}}$ entries.

saturate

Boolean indicating if values larger than $2^{\text{mostSignificantBit}-1}$ are saturated (default **TRUE**) or not (**FALSE**).

mask

Pointer to a mask to apply the function only on a particular region in the image. Note: the mask must have the same size as the source image.

EasyImage.HistogramThreshold

Determines an appropriate threshold level based on the histogram contents, using an automatic threshold mode.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void HistogramThreshold(
    Euresys.Open_eVision_2_10.EBWHistogramVector histogram,
    ref uint threshold,
    out float averageOfPixelsBelowThreshold,
    out float averageOfPixelsAboveThreshold,
    float relativeThreshold,
    uint from,
    uint to
)
```

Parameters

histogram

Pointer to a vector containing an image histogram.

threshold

reference to the threshold value. Before calling the function, must be set to the appropriate thresholding mode, as defined by [EThresholdMode](#).

averageOfPixelsBelowThreshold

Average gray level of the dark pixels (below threshold).

averageOfPixelsAboveThreshold

Average gray level of the light pixels (above threshold).

relativeThreshold

Relative threshold value, relevant only in the [Relative](#) mode.

from

Lower bound of the analyzed gray-level range.

to

Upper bound of the analyzed gray-level range.

Remarks

Additionally, returns the average gray levels in the regions below and above the threshold. The threshold level can be computed by analyzing a range of gray levels in the histogram.

EasyImage.HistogramThresholdBW16

Determines an appropriate threshold level based on the histogram contents, using an automatic threshold mode.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void HistogramThresholdBW16(
    Euresys.Open_eVision_2_10.EBWHistogramVector histogram,
    ref uint threshold,
    out float averageOfPixelsBelowThreshold,
    out float averageOfPixelsAboveThreshold,
    float relativeThreshold,
    uint from,
    uint to
)
```

Parameters

histogram

Pointer to a vector containing an image histogram.

threshold

reference to the threshold value. Before calling the function, must be set to the appropriate thresholding mode, as defined by [EThresholdMode](#).

averageOfPixelsBelowThreshold

Average gray level of the dark pixels (below threshold).

averageOfPixelsAboveThreshold

Average gray level of the light pixels (above threshold).

relativeThreshold

Relative threshold value, relevant only in the [Relative](#) mode.

from

Lower bound of the analyzed gray-level range.

to

Upper bound of the analyzed gray-level range.

Remarks

Additionally, returns the average gray levels in the regions below and above the threshold. The threshold level can be computed by analyzing a range of gray levels in the histogram.

EasyImage.HitAndMiss

Apply the morphological hit-and-miss transform to detect a particular pattern of foreground and background pixels in a BW8, BW16 or C24 image/ROI.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void HitAndMiss(
    Euresys.Open_eVision_2_10.EROIBW8 source,
    Euresys.Open_eVision_2_10.EROIBW8 destination,
    Euresys.Open_eVision_2_10.EHitAndMissKernel kernel
)

void HitAndMiss(
    Euresys.Open_eVision_2_10.EROIBW16 source,
    Euresys.Open_eVision_2_10.EROIBW16 destination,
    Euresys.Open_eVision_2_10.EHitAndMissKernel kernel
)

void HitAndMiss(
    Euresys.Open_eVision_2_10.EROIC24 source,
    Euresys.Open_eVision_2_10.EROIC24 destination,
    Euresys.Open_eVision_2_10.EHitAndMissKernel kernel
)
```

Parameters

source

The source image/ROI.

destination

The destination image/ROI.

kernel

The hit-and-miss kernel.

EasyImage.HorizontalMirror

Mirrors an image horizontally (the columns are swapped).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void HorizontalMirror(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage
)

void HorizontalMirror(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage
)

void HorizontalMirror(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

EasyImage.ImageToLineSegment

Copies the pixel values along a given line segment (arbitrarily oriented) to a vector.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ImageToLineSegment(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EBW8Vector path,
    int X0,
    int Y0,
    int X1,
    int Y1
)
```

```
void ImageToLineSegment(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EBW16Vector path,  
    int X0,  
    int Y0,  
    int X1,  
    int Y1  
)
```

```
void ImageToLineSegment(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EC24Vector path,  
    int X0,  
    int Y0,  
    int X1,  
    int Y1  
)
```

```
void ImageToLineSegment(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    Euresys.Open_eVision_2_10.EBW8 outOfMaskValue,  
    Euresys.Open_eVision_2_10.EBW8Vector path,  
    int X0,  
    int Y0,  
    int X1,  
    int Y1  
)
```

```
void ImageToLineSegment(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    Euresys.Open_eVision_2_10.EBW16 outOfMaskValue,  
    Euresys.Open_eVision_2_10.EBW16Vector path,  
    int X0,  
    int Y0,  
    int X1,  
    int Y1  
)
```



```
void ImageToLineSegment(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    Euresys.Open_eVision_2_10.EC24 outOfMaskValue,  
    Euresys.Open_eVision_2_10.EC24Vector path,  
    int X0,  
    int Y0,  
    int X1,  
    int Y1  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

path

Pointer to the destination vector.

X0

Coordinates of the starting point of the segment.

Y0

Coordinates of the starting point of the segment.

X1

Coordinates of the ending point of the segment.

Y1

Coordinates of the ending point of the segment.

mask

Pointer to a mask to apply the function only on a particular region in the image. Note: the mask must have the same size as the source image.

outOfMaskValue

The value to be given to the pixels that lie out of the mask.

Remarks

The line segment must be wholly contained within the image. The vector length is adjusted automatically.

EasyImage.ImageToPath

Given a path described by coordinates in a path vector, copies the corresponding pixel values into the same vector.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void ImageToPath(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EBW8PathVector path
)

void ImageToPath(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EBW16PathVector path
)

void ImageToPath(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EC24PathVector path
)

void ImageToPath(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    Euresys.Open_eVision_2_10.EBW8 outOfMaskValue,
    Euresys.Open_eVision_2_10.EBW8PathVector path
)

void ImageToPath(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    Euresys.Open_eVision_2_10.EBW16 outOfMaskValue,
    Euresys.Open_eVision_2_10.EBW16PathVector path
)

void ImageToPath(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    Euresys.Open_eVision_2_10.EC24 outOfMaskValue,
    Euresys.Open_eVision_2_10.EC24PathVector path
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

path

Pointer to the destination vector.

mask

Pointer to a mask to apply the function only on a particular region in the image. Note: the mask must have the same size as the source image.

outOfMaskValue

The value to be given to the pixels that lie out of the mask.

EasyImage.IsodataThreshold

Computes a suitable threshold value for a histogram.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EBW8 IsodataThreshold(
    Euresys.Open_eVision_2_10.EBWHistogramVector histogram,
    uint from,
    uint to
)
```

Parameters

histogram

Pointer to a vector containing an image histogram.

from

Lower bound of the useful gray-level interval (by default, **0**).

to

Upper bound of the useful gray-level interval (by default, **255**).

Remarks

The "isodata" rule is used: if one computes the average gray level of pixels below the threshold and the average gray level of pixels above the threshold, the threshold lies exactly halfway between them. Optionally, the threshold selection can be restricted to a range of gray-level values.

It is possible that, in the automatic or relative thresholding modes, the computed threshold exceeds the dynamic range of the return type. In this case, the value is clipped to the maximum value that is representable in the return type.

EasyImage.IsodataThresholdBW16

Computes a suitable threshold value for a histogram.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EBW16 IsodataThresholdBW16(
    Euresys.Open_eVision_2_10.EBWHistogramVector histogram,
    uint from,
    uint to
)
```

Parameters

histogram

Pointer to a vector containing an image histogram.

from

Lower bound of the useful gray-level interval (by default, **0**).

to

Upper bound of the useful gray-level interval (by default, **65535**).

Remarks

The "isodata" rule is used: if one computes the average gray level of pixels below the threshold and the average gray level of pixels above the threshold, the threshold lies exactly halfway between them. Optionally, the threshold selection can be restricted to a range of gray-level values.

Returns the threshold.

EasyImage.LinearTransform

Applies a general affine transformation.

Namespace: Euresys.Open_eVision_2_10

```

[C#]
void LinearTransform(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    float Axx,
    float Axy,
    float Ax,
    float Ayx,
    float Ayy,
    float Ay,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    int interpolationBits
)

void LinearTransform(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    float Axx,
    float Axy,
    float Ax,
    float Ayx,
    float Ayy,
    float Ay,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    int interpolationBits
)

void LinearTransform(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    float Axx,
    float Axy,
    float Ax,
    float Ayx,
    float Ayy,
    float Ay,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    int interpolationBits
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

Axx

See formula below.

Axy

See formula below.

A_x

See formula below.

A_{yx}

See formula below.

A_{yy}

See formula below.

A_y

See formula below.

destinationImage

Pointer to the destination image/ROI.

interpolationBits

Number of bits of accuracy for interpolation. Allowed values are **0** (no interpolation, nearest neighbor), **4 (linear interpolation)** or **8 (cubic interpolation)**.

Remarks

An affine transformation is an important class of linear 2D geometric transformations which maps variables (e.g. pixel intensity values located at position (X_{Src}, Y_{Src}) in an input image) into new variables (e.g. (X_{Dst}, Y_{Dst}) in an output image) by applying a linear combination of translation, rotation, scaling and/or shearing (i.e. non-uniform scaling in some directions) operations.

The parameters of the [EasyImage::LinearTransform](#) function are the coefficients of the affine equations below:

$$X_{Dst} = A_{xx} X_{Src} + A_{xy} Y_{Src} + A_x$$

$$Y_{Dst} = A_{yx} X_{Src} + A_{yy} Y_{Src} + A_y$$

EasyImage.LineSegmentToImage

Copies the pixel values from a vector or a constant to the pixels of a given line segment (arbitrarily oriented).

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void LineSegmentToImage(  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,  
    Euresys.Open_eVision_2_10.EBW8 pixel,  
    int X0,  
    int Y0,  
    int X1,  
    int Y1  
)  
  
void LineSegmentToImage(  
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,  
    Euresys.Open_eVision_2_10.EBW16 pixel,  
    int X0,  
    int Y0,  
    int X1,  
    int Y1  
)  
  
void LineSegmentToImage(  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,  
    Euresys.Open_eVision_2_10.EC24 pixel,  
    int X0,  
    int Y0,  
    int X1,  
    int Y1  
)  
  
void LineSegmentToImage(  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,  
    Euresys.Open_eVision_2_10.EBW8Vector path,  
    int X0,  
    int Y0,  
    int X1,  
    int Y1  
)  
  
void LineSegmentToImage(  
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,  
    Euresys.Open_eVision_2_10.EBW16Vector path,  
    int X0,  
    int Y0,  
    int X1,  
    int Y1  
)
```

```
void LineSegmentToImage (  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,  
    Euresys.Open_eVision_2_10.EC24Vector path,  
    int X0,  
    int Y0,  
    int X1,  
    int Y1  
)
```

Parameters

destinationImage

Pointer to the destination image/ROI.

pixel

Constant color value.

X0

Coordinates of the starting point of the segment.

Y0

Coordinates of the starting point of the segment.

X1

Coordinates of the ending point of the segment.

Y1

Coordinates of the ending point of the segment.

path

Pointer to the source vector.

Remarks

The line segment must be wholly contained within the image.

EasyImage.LocalAverage

Computes the average in a rectangular window centered on every pixel.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
void LocalAverage(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfWidth,
    uint halfHeight
)

void LocalAverage(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    uint halfWidth,
    uint halfHeight
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI.

halfWidth

Half of the window width minus one.

halfHeight

Half of the window height minus one.

Remarks

The window dimensions can be an arbitrary odd integer.

The running time of this function does not depend on the window size.

EasyImage.LocalDeviation

Computes the standard deviation in a rectangular window centered on every pixel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void LocalDeviation(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfWidth,
    uint halfHeight
)

void LocalDeviation(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    uint halfWidth,
    uint halfHeight
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. Must not be the same as the source image.

halfWidth

Half of the window width minus one.

halfHeight

Half of the window height minus one.

Remarks

The window dimensions can be an arbitrary odd integer.

The running time of this function does not depend on the window size.

EasyImage.Lut

Transforms the gray levels of an image, using a lookup table stored in a vector (of unsigned values).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Lut(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    Euresys.Open_eVision_2_10.EBW16Vector lookupTable
)

void Lut(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    Euresys.Open_eVision_2_10.EBW8Vector lookupTable
)

void Lut(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    Euresys.Open_eVision_2_10.EBW8Vector lookupTable,
    uint numberOfScalingBits
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI.

lookupTable

Pointer to the lookup vector.

numberOfScalingBits

Number of scaling bits (or right padding bits).

Remarks

A 16-bit image usually does not make use of its 16 bits. In most cases, only 10 or 12 bits are used. These bits are called *significant bits*. In the 16-bit information, significant bits can be left aligned, right aligned or not aligned at all. To indicate which are the significant bits, we have to tell how many bits are significant and the number of right padding bits (0 right padding bit means that significant bits are right aligned).

The number of significant bits is given by the number of Look Up table entries. For example a Lut of 1024 entries is used for an image of 10 significant bits (as $2^{10} = 1024$).

The number of right padding bits is given by means of the **numberOfScalingBits** parameter. Leaving this parameter undefined indicates that the significant bits are left aligned on the word.

EasyImage.MatchFrames

Determines the optimal shift amplitude by comparing two successive lines of the image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void MatchFrames (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    int fixedRow,
    int minimumOffset,
    int maximumOffset,
    ref int bestOffset
)

void MatchFrames (
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    int fixedRow,
    int minimumOffset,
    int maximumOffset,
    ref int bestOffset
)

void MatchFrames (
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    int fixedRow,
    int minimumOffset,
    int maximumOffset,
    ref int bestOffset
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

fixedRow

Index of the line used for comparison. Line **fixedRow** remains in place and is compared with line (**fixedRow + 1**), shifted by some amount.

minimumOffset

Minimum value of the allowed offset (positive to the right).

maximumOffset

Maximum value of the allowed offset (positive to the right).

bestOffset

Estimated shift amplitude.

Remarks

These lines should be chosen such that they cross some edges or non-uniform areas.

When an image is interlaced, the two frames (even and odd lines) are not recorded at the same time. If there is movement in the scene, a visible artifact can result (the edges of objects exhibit a "comb" effect).

When the movement is uniform and horizontal (objects on a conveyor belt), one cure to this problem is to shift one of the frames horizontally with respect to the other frame (using [EasyImage::RealignFrame](#)). The amplitude of the shift can be estimated automatically.

EasyImage.Median

Applies a median filter to an image (median of the gray values in a 3x3 neighborhood).

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void Median(
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW1 destinationImage
)

void Median(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage
)

void Median(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage
)

void Median(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. Must not be the same as source image.

EasyImage.ModulusImage

Prepares a lookup-table image for use for gradient magnitude computation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ModulusImage (
    Euresys.Open_eVision_2_10.EImageBW8 destinationImage,
    float gain
)
```

Parameters

destinationImage

Pointer to the destination image.

gain

Gain value to be applied to the modulus. **1** saturates; **1/Sqrt(2)** does not.

Remarks

The modulus of the gradient argument can be adjusted to avoid saturation. [EasyImage::ModulusImage](#) sets a lookup-table image for use with function [EasyImage::GradientScalar](#), ready to compute the modulus of the gradient in the source image, i.e. its amplitude (as defined by the Euclidian norm). The argument will be returned as a value in range **0..255** suitable for storage in an [EImageBW8](#) or as a value in range **0..65535** suitable for storage in an [EImageBW16](#). A gain coefficient can be adjusted to avoid saturation (gain = 1 saturates gradient amplitudes larger than 255 in the [EBW8](#) case and 65535 in the [EBW16](#) case; gain = **1/Sqrt(2)** never saturates).

EasyImage.MorphoGradientBox

Computes the morphological gradient of an image using a rectangular kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void MorphoGradientBox(
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW1 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)

void MorphoGradientBox(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)

void MorphoGradientBox(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)

void MorphoGradientBox(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

halfOfKernelWidth

Half of the box width minus one (by default, **halfOfKernelWidth = 1**; **0** is allowed).

halfOfKernelHeight

Half of the box height minus one (by default, same as **halfOfKernelWidth**; **0** is allowed).

Remarks

The morphological gradient is the difference between the dilation and the erosion of the image, using the same structuring element.

The kernel size is a pair of odd numbers; they must be halved before they are passed. For instance, a 3x5 kernel is passed as 1x2.

EasyImage.MorphoGradientDisk

Computes the morphological gradient of an image using a quasi-circular kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void MorphoGradientDisk(
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW1 destinationImage,
    uint halfOfKernelWidth
)

void MorphoGradientDisk(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth
)

void MorphoGradientDisk(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    uint halfOfKernelWidth
)
```



```
void MorphoGradientDisk(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,  
    uint halfOfKernelWidth  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

halfOfKernelWidth

Half width of the kernel minus one (by default, **halfOfKernelWidth = 1**; **0** is allowed).

Remarks

The morphological gradient is the difference between the dilation and the erosion of the image, using the same structuring element.

EasyImage.Normalize

Normalizes an image, i.e. applies a linear transform to the gray levels so that their average and standard deviation are imposed.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void Normalize(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,  
    float imposedAverage,  
    float imposedStandardDeviation  
)
```

```
void Normalize(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,  
    float imposedAverage,  
    float imposedStandardDeviation  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI.

imposedAverage

Imposed average.

imposedStandardDeviation

Imposed standard deviation.

EasyImage.Offset

Transforms an image, applying a gain and offset to all pixels.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void Offset(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,  
    Euresys.Open_eVision_2_10.EColor Offset  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI.

Offset

Constant offset. By default (argument omitted) **0**, i.e. no change.

Remarks

The gain should remain close to **1**, and allows contrast adjustment of the image.

The offset can be positive or negative, and allows to adjust the image intensity. The resulting values are always saturated to range **[0..255]**.

For color images, the separate gain and offset values are specified as triple of values stored in a [EColor](#). The default values leave the image unchanged.

Internally, the computations are achieved through fixed-point arithmetic with 5 bits of precision for the fractional part. This can result in loss of precision with small gains.

EasyImage.OpenBox

Performs an opening on an image (erosion followed by dilation) on a rectangular kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void OpenBox(
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW1 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)

void OpenBox(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)
```

```

void OpenBox(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)

void OpenBox(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

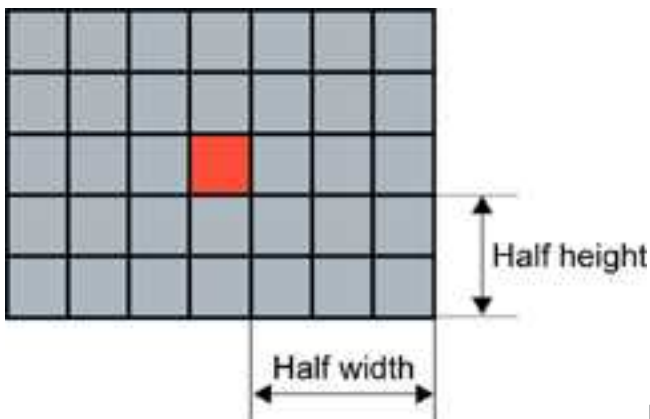
halfOfKernelWidth

Half of the box width minus one, as shown on the picture below (by default, **halfOfKernelWidth = 1**; **0** is allowed).

halfOfKernelHeight

Half of the box height minus one, as shown on the picture below (by default, same as **halfOfKernelWidth**; **0** is allowed).

Remarks



Rectangular kernel of half width = 3 and half height = 2

EasyImage.OpenDisk

Performs an opening on an image (erosion followed by dilation) on a quasi-circular kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void OpenDisk(
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW1 destinationImage,
    uint halfOfKernelWidth
)

void OpenDisk(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth
)

void OpenDisk(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    uint halfOfKernelWidth
)

void OpenDisk(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    uint halfOfKernelWidth
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. If **NULL** (default), this operation is destructive (i.e. applied to the source image).

halfOfKernelWidth

Half width of the kernel minus one, as shown on the picture below (by default, **halfOfKernelWidth = 1**; **0** is allowed).

EasyImage.Oper

Applies the desired arithmetic or logic pixel-wise operator between two images or constants.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void Oper(
    Euresys.Open_eVision_2_10.EArithmeticLogicOperation operation,
    Euresys.Open_eVision_2_10.EBW1 constant,
    Euresys.Open_eVision_2_10.EROIBW1 destinationImage
)

void Oper(
    Euresys.Open_eVision_2_10.EArithmeticLogicOperation operation,
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW1 destinationImage
)

void Oper(
    Euresys.Open_eVision_2_10.EArithmeticLogicOperation operation,
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage0,
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage1,
    Euresys.Open_eVision_2_10.EROIBW1 destinationImage
)

void Oper(
    Euresys.Open_eVision_2_10.EArithmeticLogicOperation operation,
    Euresys.Open_eVision_2_10.EBW8 constant,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage
)

void Oper(
    Euresys.Open_eVision_2_10.EArithmeticLogicOperation operation,
    Euresys.Open_eVision_2_10.EBW16 constant,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage
)
```

```
void Oper(  
    Euresys.Open_eVision_2_10.EArithmeticLogicOperation operation,  
    Euresys.Open_eVision_2_10.EC24 constant,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage  
)  
  
void Oper(  
    Euresys.Open_eVision_2_10.EArithmeticLogicOperation operation,  
    Euresys.Open_eVision_2_10.EBW8 constant,  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage  
)  
  
void Oper(  
    Euresys.Open_eVision_2_10.EArithmeticLogicOperation operation,  
    Euresys.Open_eVision_2_10.EBW16 constant,  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage  
)  
  
void Oper(  
    Euresys.Open_eVision_2_10.EArithmeticLogicOperation operation,  
    Euresys.Open_eVision_2_10.EC24 constant,  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage  
)  
  
void Oper(  
    Euresys.Open_eVision_2_10.EArithmeticLogicOperation operation,  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EBW8 constant,  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage  
)  
  
void Oper(  
    Euresys.Open_eVision_2_10.EArithmeticLogicOperation operation,  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EBW16 constant,  
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage  
)
```

```

void Oper(
    Euresys.Open_eVision_2_10.EArithmeticLogicOperation operation,
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EC24 constant,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage
)

void Oper(
    Euresys.Open_eVision_2_10.EArithmeticLogicOperation operation,
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage
)

void Oper(
    Euresys.Open_eVision_2_10.EArithmeticLogicOperation operation,
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage
)

void Oper(
    Euresys.Open_eVision_2_10.EArithmeticLogicOperation operation,
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage
)

void Oper(
    Euresys.Open_eVision_2_10.EArithmeticLogicOperation operation,
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage
)

void Oper(
    Euresys.Open_eVision_2_10.EArithmeticLogicOperation operation,
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage0,
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage1,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage
)

void Oper(
    Euresys.Open_eVision_2_10.EArithmeticLogicOperation operation,
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage0,
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage1,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage
)

```



```

void Oper(
    Euresys.Open_eVision_2_10.EArithmeticLogicOperation operation,
    Euresys.Open_eVision_2_10.EROIC24 sourceImage0,
    Euresys.Open_eVision_2_10.EROIC24 sourceImage1,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage
)

void Oper(
    Euresys.Open_eVision_2_10.EArithmeticLogicOperation operation,
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage0,
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage1,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage
)

void Oper(
    Euresys.Open_eVision_2_10.EArithmeticLogicOperation operation,
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage0,
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage1,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage
)

void Oper(
    Euresys.Open_eVision_2_10.EArithmeticLogicOperation operation,
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage0,
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage1,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage
)

void Oper(
    Euresys.Open_eVision_2_10.EArithmeticLogicOperation operation,
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage0,
    Euresys.Open_eVision_2_10.EROIC24 sourceImage1,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage
)

void Oper(
    Euresys.Open_eVision_2_10.EArithmeticLogicOperation operation,
    Euresys.Open_eVision_2_10.EROIC24 sourceImage0,
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage1,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage
)

```

Parameters

operation

Arithmetic or logic operator, as defined by [EArithmeticLogicOperation](#).

constant

Gray-level or color constant.

destinationImage

Pointer to the destination image/ROI.

sourceImage

Pointer to the second source image/ROI (right operand).

sourceImage0

Pointer to the first source image/ROI (left operand).

sourceImage1

Pointer to the second source image/ROI (right operand).

Remarks

The source and destination images may be the same.

When the source operands are two color images/constants, the components are combined pair-wise. The result is a color image.

When the source operands are a color image and a gray-level image, each color component is combined with the gray-level component. The result is a color image.

EasyImage.Overlay

Overlays an image on the top of a color image, at a given position.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Overlay(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC16 destinationImage,
    int panX,
    int panY,
    Euresys.Open_eVision_2_10.EC24 referenceValue
)
```

```
void Overlay(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC15 destinationImage,  
    int panX,  
    int panY,  
    Euresys.Open_eVision_2_10.EC24 referenceValue  
)
```

```
void Overlay(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,  
    int panX,  
    int panY,  
    Euresys.Open_eVision_2_10.EC24 referenceValue  
)
```

```
void Overlay(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    Euresys.Open_eVision_2_10.EROIC15 destinationImage,  
    int panX,  
    int panY  
)
```

```
void Overlay(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    Euresys.Open_eVision_2_10.EROIC16 destinationImage,  
    int panX,  
    int panY  
)
```

```
void Overlay(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,  
    int panX,  
    int panY  
)
```

```
void Overlay(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 overlay,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,  
    int panX,  
    int panY,  
    Euresys.Open_eVision_2_10.EC24 referenceValue  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI.

panX

Translation for panning in the horizontal direction, in pixels.

panY

Translation for panning in the vertical direction, in pixels.

referenceValue

Reference color.

mask

Pointer to a mask to apply the function only on a particular region in the image. Note: the mask must have the same size as the source image.

overlay

When a **BW8** source image is specified, pointer to the overlay image/ROI.

Remarks

If a color image is provided as the source image, all the pixels of this image are copied to the destination image, but the ones that equal the reference color.

If a **BW8** image is provided as the source image, all the pixels of the overlay image are copied to the destination image, but the ones that equal the reference color, the latter being replaced by the content of the source image.

EasyImage.OverlayColor

Gets/Sets the color of the overlay in the destination image when a **BW8** Image is used as overlay source image in functions.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
static Euresys.Open_eVision_2_10.EC24 OverlayColor
    { get; set; }
```

Remarks

Note. When a **C24** Image is used as overlay source image, the color of the overlay in destination image is the same as the one in the overlay source image, thus allowing multi colored overlays.

EasyImage.PathToImage

Given a path described by coordinates in a path vector, copies the pixel values from the path vector to the corresponding image pixels.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void PathToImage (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EBW8PathVector path
)

void PathToImage (
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EBW16PathVector path
)

void PathToImage (
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EC24PathVector path
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

path

Pointer to the destination vector.

EasyImage.PixelAverage

Computes the average pixel value in a gray-level or color image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void PixelAverage(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    out float average
)

void PixelAverage(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    out float average
)

void PixelAverage(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    out float average0,
    out float average1,
    out float average2
)

void PixelAverage(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    out float average
)

void PixelAverage(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    out float average
)
```

```

void PixelAverage (
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    out float average0,
    out float average1,
    out float average2
)

void PixelAverage (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 inputMask,
    out float average
)

void PixelAverage (
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 inputMask,
    out float average
)

void PixelAverage (
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 inputMask,
    out float average0,
    out float average1,
    out float average2
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

average

Reference to the average gray-level value.

average0

Reference to the average values for the first color channel.

average1

Reference to the average values for the second color channel.

average2

Reference to the average values for the third color channel.

region

Pointer to a region to apply the function only on a particular region in the image.

inputMask

Pointer to the mask, which allows functions to be applied on a particular region in the image.

EasyImage.PixelCompare

Counts the number of pixels differing between two images.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
uint PixelCompare(  
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage0,  
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage1  
)  
  
uint PixelCompare(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage0,  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage1  
)  
  
uint PixelCompare(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage0,  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage1  
)  
  
uint PixelCompare(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage0,  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage1  
)  
  
uint PixelCompare(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage0,  
    Euresys.Open_eVision_2_10.ERegion region,  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage1  
)  
  
uint PixelCompare(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage0,  
    Euresys.Open_eVision_2_10.ERegion region,  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage1  
)
```



```

uint PixelCompare(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage0,
    Euresys.Open_eVision_2_10.ERegion region,
    Euresys.Open_eVision_2_10.EROIC24 sourceImage1
)

uint PixelCompare(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage0,
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage1,
    Euresys.Open_eVision_2_10.EROIBW8 mask
)

uint PixelCompare(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage0,
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage1,
    Euresys.Open_eVision_2_10.EROIBW8 mask
)

uint PixelCompare(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage0,
    Euresys.Open_eVision_2_10.EROIC24 sourceImage1,
    Euresys.Open_eVision_2_10.EROIBW8 mask
)

```

Parameters

sourceImage0

Pointer to the first source image/ROI.

sourceImage1

Pointer to the second source image/ROI.

region

Pointer to a region to apply the function only on a particular region in the image.

mask

Pointer to a mask to apply the function only on a particular region in the image. Note: the mask must have the same size as the source image.

EasyImage.PixelCount

Counts the pixels in the three value classes separated by two thresholds.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void PixelCount(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EBW8 lowThreshold,  
    Euresys.Open_eVision_2_10.EBW8 highThreshold,  
    out int numberOfPixelsBelowThreshold,  
    out int numberOfPixelsBetweenThresholds,  
    out int numberOfPixelsAboveThreshold  
)
```

```
void PixelCount(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EBW16 lowThreshold,  
    Euresys.Open_eVision_2_10.EBW16 highThreshold,  
    out int numberOfPixelsBelowThreshold,  
    out int numberOfPixelsBetweenThresholds,  
    out int numberOfPixelsAboveThreshold  
)
```

```
void PixelCount(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.ERegion region,  
    Euresys.Open_eVision_2_10.EBW8 lowThreshold,  
    Euresys.Open_eVision_2_10.EBW8 highThreshold,  
    out int numberOfPixelsBelowThreshold,  
    out int numberOfPixelsBetweenThresholds,  
    out int numberOfPixelsAboveThreshold  
)
```

```
void PixelCount(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.ERegion region,  
    Euresys.Open_eVision_2_10.EBW16 lowThreshold,  
    Euresys.Open_eVision_2_10.EBW16 highThreshold,  
    out int numberOfPixelsBelowThreshold,  
    out int numberOfPixelsBetweenThresholds,  
    out int numberOfPixelsAboveThreshold  
)
```

```
void PixelCount(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.ERegion region,  
    Euresys.Open_eVision_2_10.EBW8 lowThreshold,  
    Euresys.Open_eVision_2_10.EBW8 highThreshold,  
    out System.UInt64 numberOfPixelsBelowThreshold,  
    out System.UInt64 numberOfPixelsBetweenThresholds,  
    out System.UInt64 numberOfPixelsAboveThreshold  
)
```

```
void PixelCount(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.ERegion region,  
    Euresys.Open_eVision_2_10.EBW16 lowThreshold,  
    Euresys.Open_eVision_2_10.EBW16 highThreshold,  
    out System.UInt64 numberOfPixelsBelowThreshold,  
    out System.UInt64 numberOfPixelsBetweenThresholds,  
    out System.UInt64 numberOfPixelsAboveThreshold  
)
```

```
void PixelCount(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EBW8 lowThreshold,  
    Euresys.Open_eVision_2_10.EBW8 highThreshold,  
    out System.UInt64 numberOfPixelsBelowThreshold,  
    out System.UInt64 numberOfPixelsBetweenThresholds,  
    out System.UInt64 numberOfPixelsAboveThreshold  
)
```

```
void PixelCount(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EBW16 lowThreshold,  
    Euresys.Open_eVision_2_10.EBW16 highThreshold,  
    out System.UInt64 numberOfPixelsBelowThreshold,  
    out System.UInt64 numberOfPixelsBetweenThresholds,  
    out System.UInt64 numberOfPixelsAboveThreshold  
)
```

```

void PixelCount(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    Euresys.Open_eVision_2_10.EBW8 lowThreshold,
    Euresys.Open_eVision_2_10.EBW8 highThreshold,
    out int numberOfPixelsBelowThreshold,
    out int numberOfPixelsBetweenThresholds,
    out int numberOfPixelsAboveThreshold
)

void PixelCount(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    Euresys.Open_eVision_2_10.EBW16 lowThreshold,
    Euresys.Open_eVision_2_10.EBW16 highThreshold,
    out int numberOfPixelsBelowThreshold,
    out int numberOfPixelsBetweenThresholds,
    out int numberOfPixelsAboveThreshold
)

void PixelCount(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    Euresys.Open_eVision_2_10.EBW8 lowThreshold,
    Euresys.Open_eVision_2_10.EBW8 highThreshold,
    out System.UInt64 numberOfPixelsBelowThreshold,
    out System.UInt64 numberOfPixelsBetweenThresholds,
    out System.UInt64 numberOfPixelsAboveThreshold
)

void PixelCount(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    Euresys.Open_eVision_2_10.EBW16 lowThreshold,
    Euresys.Open_eVision_2_10.EBW16 highThreshold,
    out System.UInt64 numberOfPixelsBelowThreshold,
    out System.UInt64 numberOfPixelsBetweenThresholds,
    out System.UInt64 numberOfPixelsAboveThreshold
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

lowThreshold

Inferior threshold.

highThreshold

Superior threshold.

numberOfPixelsBelowThreshold

Reference to the count of pixels strictly below the inferior threshold.

numberOfPixelsBetweenThresholds

Reference to the count of pixels above or equal to the inferior threshold, and strictly below the superior threshold.

numberOfPixelsAboveThreshold

Reference to the count of pixels above or equal to the superior threshold.

region

Pointer to a region to apply the function only on a particular region in the image.

mask

Pointer to a mask to apply the function only on a particular region in the image. Note: the mask must have the same size as the source image.

EasyImage.PixelMax

Computes the maximum gray-level value in an image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void PixelMax(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    out Euresys.Open_eVision_2_10.EBW8 maximumValue
)

void PixelMax(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    out Euresys.Open_eVision_2_10.EBW16 maximumValue
)

void PixelMax(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    out Euresys.Open_eVision_2_10.EBW8 maximumValue
)
```

```
void PixelMax(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.ERegion region,  
    out Euresys.Open_eVision_2_10.EBW16 maximumValue  
)  
  
void PixelMax(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    out Euresys.Open_eVision_2_10.EBW8 maximumValue  
)  
  
void PixelMax(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    out Euresys.Open_eVision_2_10.EBW16 maximumValue  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

maximumValue

Reference to the maximum value.

region

Pointer to a region to apply the function only on a particular region in the image.

mask

Pointer to a mask to apply the function only on a particular region in the image. Note: the mask must have the same size as the source image.

EasyImage.PixelMaxBW16

Computes the maximum gray-level value in an image.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void PixelMaxBW16(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    out Euresys.Open_eVision_2_10.EBW16 maximumValue  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

maximumValue

Reference to the maximum value.

EasyImage.PixelMaxBW8

Computes the maximum gray-level value in an image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void PixelMaxBW8(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    out Euresys.Open_eVision_2_10.EBW8 maximumValue  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

maximumValue

Reference to the maximum value.

EasyImage.PixelMin

Computes the minimum gray-level value in an image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void PixelMin(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    out Euresys.Open_eVision_2_10.EBW8 minimumValue
)

void PixelMin(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    out Euresys.Open_eVision_2_10.EBW16 minimumValue
)

void PixelMin(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    out Euresys.Open_eVision_2_10.EBW8 minimumValue
)

void PixelMin(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    out Euresys.Open_eVision_2_10.EBW16 minimumValue
)

void PixelMin(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    out Euresys.Open_eVision_2_10.EBW8 minimumValue
)

void PixelMin(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    out Euresys.Open_eVision_2_10.EBW16 minimumValue
)
```


Parameters

sourceImage

Pointer to the source image/ROI.

minimumValue

Reference to the minimum value.

region

Pointer to a region to apply the function only on a particular region in the image.

mask

Pointer to a mask to apply the function only on a particular region in the image. Note: the mask must have the same size as the source image.

EasyImage.PixelMinBW16

Computes the minimum gray-level value in an image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void PixelMinBW16(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    out Euresys.Open_eVision_2_10.EBW16 minimumValue
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

minimumValue

Reference to the minimum value.

EasyImage.PixelMinBW8

Computes the minimum gray-level value in an image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void PixelMinBW8 (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    out Euresys.Open_eVision_2_10.EBW8 minimumValue
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

minimumValue

Reference to the minimum value.

EasyImage.PixelStat

Computes the minimum, maximum and average gray-level values in an image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void PixelStat (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    out Euresys.Open_eVision_2_10.EBW8 minimumValue,
    out Euresys.Open_eVision_2_10.EBW8 maximumValue,
    out float average
)

void PixelStat (
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    out Euresys.Open_eVision_2_10.EBW16 minimumValue,
    out Euresys.Open_eVision_2_10.EBW16 maximumValue,
    out float average
)
```

```

void PixelStat(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    out Euresys.Open_eVision_2_10.EBW8 minimumValue,
    out Euresys.Open_eVision_2_10.EBW8 maximumValue,
    out float average
)

void PixelStat(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    out Euresys.Open_eVision_2_10.EBW16 minimumValue,
    out Euresys.Open_eVision_2_10.EBW16 maximumValue,
    out float average
)

void PixelStat(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    out Euresys.Open_eVision_2_10.EBW8 minimumValue,
    out Euresys.Open_eVision_2_10.EBW8 maximumValue,
    out float average
)

void PixelStat(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    out Euresys.Open_eVision_2_10.EBW16 minimumValue,
    out Euresys.Open_eVision_2_10.EBW16 maximumValue,
    out float average
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

minimumValue

Reference to the minimum value.

maximumValue

Reference to the maximum value.

average

Reference to the average value.

region

Pointer to a region to apply the function only on a particular region in the image.

mask

Pointer to a mask to apply the function only on a particular region in the image. Note: the mask must have the same size as the source image.

EasyImage.PixelStatBW16

Computes the minimum, maximum and average gray-level values in an image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void PixelStatBW16(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    out Euresys.Open_eVision_2_10.EBW16 minimumValue,
    out Euresys.Open_eVision_2_10.EBW16 maximumValue,
    out float average
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

minimumValue

Reference to the minimum value.

maximumValue

Reference to the maximum value.

average

Reference to the average value.

EasyImage.PixelStatBW8

Computes the minimum, maximum and average gray-level values in an image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void PixelStatBW8 (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    out Euresys.Open_eVision_2_10.EBW8 minimumValue,
    out Euresys.Open_eVision_2_10.EBW8 maximumValue,
    out float average
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

minimumValue

Reference to the minimum value.

maximumValue

Reference to the maximum value.

average

Reference to the average value.

EasyImage.PixelStdDev

Computes the average gray-level or color value in an image, the standard deviation of the color components, and the correlation between the color components (in the case of color images).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void PixelStdDev (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    out float standardDeviation,
    out float mean
)
```

```
void PixelStdDev(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    out float standardDeviation,  
    out float mean  
)  
  
void PixelStdDev(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    out float standardDeviation0,  
    out float standardDeviation1,  
    out float standardDeviation2,  
    out float correlation01,  
    out float correlation12,  
    ref float correlation20,  
    out float mean0,  
    out float mean1,  
    out float mean2  
)  
  
void PixelStdDev(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.ERegion region,  
    out float standardDeviation,  
    out float mean  
)  
  
void PixelStdDev(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.ERegion region,  
    out float standardDeviation,  
    out float mean  
)
```

```

void PixelStdDev(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    out float standardDeviation0,
    out float standardDeviation1,
    out float standardDeviation2,
    out float correlation01,
    out float correlation12,
    ref float correlation20,
    out float mean0,
    out float mean1,
    out float mean2
)

void PixelStdDev(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    out float standardDeviation,
    out float mean
)

void PixelStdDev(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    out float standardDeviation,
    out float mean
)

void PixelStdDev(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    out float standardDeviation0,
    out float standardDeviation1,
    out float standardDeviation2,
    out float correlation01,
    out float correlation12,
    ref float correlation20,
    out float mean0,
    out float mean1,
    out float mean2
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

standardDeviation

Reference to a variable in which the standard deviation of the pixel values is to be stored (for gray-level images).

mean

Reference to a variable in which the average value of the pixels is to be stored (for gray-level images).

standardDeviation0

Reference to a variable in which the standard deviation of the values of the first color component is to be stored (for color images).

standardDeviation1

Reference to a variable in which the standard deviation of the values of the second color component is to be stored (for color images).

standardDeviation2

Reference to a variable in which the standard deviation of the values of the third color component is to be stored (for color images).

correlation01

Reference to a variable in which the correlation between the values of the first color component and the second color component is to be stored (for color images).

correlation12

Reference to a variable in which the correlation between the values of the second color component and the third color component is to be stored (for color images).

correlation20

-

mean0

Reference to a variable in which the average value of the first color component is to be stored (for color images).

mean1

Reference to a variable in which the average value of the second color component is to be stored (for color images).

mean2

Reference to a variable in which the average value of the third color component is to be stored (for color images).

region

Pointer to a region to apply the function only on a particular region in the image.

mask

Pointer to a mask to apply the function only on a particular region in the image. Note: the mask must have the same size as the source image.

Remarks

The variance can be obtained from the standard deviation by squaring it.

EasyImage.PixelVariance

For a gray-level image, computes the mean and variance of the pixel values.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void PixelVariance(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    out float variance,
    out float mean
)

void PixelVariance(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    out float variance,
    out float mean
)

void PixelVariance(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    out float variance0,
    out float variance1,
    out float variance2,
    out float covariance01,
    out float covariance12,
    out float covariance20,
    out float mean0,
    out float mean1,
    out float mean2
)

void PixelVariance(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    out float variance,
    out float mean
)
```

```
void PixelVariance(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.ERegion region,  
    out float variance,  
    out float mean  
)
```

```
void PixelVariance(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.ERegion region,  
    out float variance0,  
    out float variance1,  
    out float variance2,  
    out float covariance01,  
    out float covariance12,  
    out float covariance20,  
    out float mean0,  
    out float mean1,  
    out float mean2  
)
```

```
void PixelVariance(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    out float variance,  
    out float mean  
)
```

```
void PixelVariance(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    out float variance,  
    out float mean  
)
```

```
void PixelVariance(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    out float variance0,  
    out float variance1,  
    out float variance2,  
    out float covariance01,  
    out float covariance12,  
    out float covariance20,  
    out float mean0,  
    out float mean1,  
    out float mean2  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

variance

Reference to the covariances of the pairs of pixel component values.

mean

Reference to the mean pixel component values.

variance0

Reference to the covariances of the pairs of pixel component values.

variance1

Reference to the covariances of the pairs of pixel component values.

variance2

Reference to the covariances of the pairs of pixel component values.

covariance01

Reference to the covariances of the pairs of pixel component values.

covariance12

Reference to the covariances of the pairs of pixel component values.

covariance20

Reference to the covariances of the pairs of pixel component values.

mean0

Reference to the mean pixel component values.

mean1

Reference to the mean pixel component values.

mean2

Reference to the mean pixel component values.

region

Pointer to a region to apply the function only on a particular region in the image.

mask

Pointer to a mask to apply the function only on a particular region in the image. Note: the mask must have the same size as the source image.

Remarks

For a color image, computes the means of the three pixel color components, the variances of the components and the covariances between pairs of components.

EasyImage.ProfileDerivative

Computes the first derivative of a profile extracted from a gray-level image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ProfileDerivative(
    Euresys.Open_eVision_2_10.EBW8Vector sourceVector,
    Euresys.Open_eVision_2_10.EBW8Vector destinationVector
)

void ProfileDerivative(
    Euresys.Open_eVision_2_10.EBW16Vector sourceVector,
    Euresys.Open_eVision_2_10.EBW16Vector destinationVector
)
```

Parameters

sourceVector

Pointer to the source vector.

destinationVector

Pointer to the destination vector.

Remarks

Taking the derivative transforms transitions (edges) into peaks.

Note. Since the [EBW8](#) data type only handles unsigned values, the derivative is shifted up by 128. Values under [above] 128 correspond to negative [positive] derivative (decreasing [increasing] slope).

EasyImage.ProjectOnAColumn

Projects an image horizontally onto a column.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void ProjectOnAColumn(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EBW32Vector destinationVector  
)  
  
void ProjectOnAColumn(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EBW32Vector destinationVector  
)  
  
void ProjectOnAColumn(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EBW8Vector destinationVector  
)  
  
void ProjectOnAColumn(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EBW16Vector destinationVector  
)  
  
void ProjectOnAColumn(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EC24Vector destinationVector  
)
```

```

void ProjectOnAColumn(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    Euresys.Open_eVision_2_10.EBW32Vector destinationVector
)

void ProjectOnAColumn(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    Euresys.Open_eVision_2_10.EBW32Vector destinationVector
)

void ProjectOnAColumn(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    Euresys.Open_eVision_2_10.EBW8Vector destinationVector
)

void ProjectOnAColumn(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    Euresys.Open_eVision_2_10.EBW16Vector destinationVector
)

void ProjectOnAColumn(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    Euresys.Open_eVision_2_10.EC24Vector destinationVector
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationVector

Pointer to the destination vector.

mask

Pointer to a mask to apply the function only on a particular region in the image. Note: the mask must have the same size as the source image.

Remarks

Pixel gray/color levels are added when projecting into an [EBW32Vector](#). When projecting into an [EBW8Vector](#)/[EBW16Vector](#)/[EC24Vector](#), pixel values are averaged, instead.

EasyImage.ProjectOnARow

Projects an image vertically onto a row.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void ProjectOnARow(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EBW32Vector destinationVector  
)  
  
void ProjectOnARow(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EBW32Vector destinationVector  
)  
  
void ProjectOnARow(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EBW8Vector destinationVector  
)  
  
void ProjectOnARow(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EBW16Vector destinationVector  
)  
  
void ProjectOnARow(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EC24Vector destinationVector  
)  
  
void ProjectOnARow(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    Euresys.Open_eVision_2_10.EBW32Vector destinationVector  
)
```

```

void ProjectOnARow(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    Euresys.Open_eVision_2_10.EBW32Vector destinationVector
)

void ProjectOnARow(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    Euresys.Open_eVision_2_10.EBW8Vector destinationVector
)

void ProjectOnARow(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    Euresys.Open_eVision_2_10.EBW16Vector destinationVector
)

void ProjectOnARow(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    Euresys.Open_eVision_2_10.EC24Vector destinationVector
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationVector

Pointer to the destination vector.

mask

Pointer to a mask to apply the function only on a particular region in the image. Note: the mask must have the same size as the source image.

Remarks

Pixel gray/color levels are added when projecting into an [EBW32Vector](#). When projecting into an [EBW8Vector](#)/[EBW16Vector](#)/[EC24Vector](#), pixel values are averaged, instead.

EasyImage.RealignFrame

Shifts one frame of the image horizontally.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void RealignFrame (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    int offset,
    uint fixedRow
)

void RealignFrame (
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    int offset,
    uint fixedRow
)

void RealignFrame (
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    int offset,
    uint fixedRow
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI.

offset

Indicates the number of pixels by which to shift (positive to the right).

fixedRow

Specifies which frame remains unchanged (the frame made up of all lines of the same parity as **fixedRow**; by default, **fixedRow = 0**).

Remarks

The same image should be used as source and destination. If the destination image differs from the source image, only the shifted rows are copied. To use a different destination image, the source image must be copied first in the destination image object.

When an image is interlaced, the two frames (even and odd lines) are not recorded at the same time. If there is movement in the scene, a visible artifact can result (the edges of objects exhibit a "comb" effect).

When the movement is uniform and horizontal (objects on a conveyor belt), one cure to this problem is to shift one of the frames horizontally with respect to the other frame. The amplitude of the shift can be estimated automatically (using [EasyImage::MatchFrames](#)).

EasyImage.RebuildFrame

Rebuilds one frame of the image by interpolation between the lines of the other frame.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void RebuildFrame (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint fixedRow
)

void RebuildFrame (
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    uint fixedRow
)

void RebuildFrame (
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    uint fixedRow
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI.

fixedRow

Specifies which frame remains unchanged (the frame made up of all lines of the same parity as **fixedRow**; by default, **fixedRow = 0**).

Remarks

The same image should be used as source and destination. If the destination image differs from the source image, only the shifted rows are copied. To use a different destination image, the source image must be copied first in the destination image object. When an image is interlaced, the two frames (even and odd lines) are not recorded at the same time. If there is movement in the scene, a visible artifact can result (the edges of objects exhibit a "comb" effect). One cure to this problem is to replace one of the frames by linearly interpolating between the lines of the other frame.

EasyImage.RecursiveAverage

Applies stronger noise reduction to small variations and conversely.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void RecursiveAverage (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 store,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    Euresys.Open_eVision_2_10.EBW16Vector lookupTable
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

store

Pointer to a 16-bit work image.

destinationImage

Pointer to the destination image/ROI.

lookupTable

Pointer to the LUT vector generated by a call to [EasyImage::SetRecursiveAverageLUT](#).

Remarks

Recursive averaging is a well known process for noise reduction by temporal integration. The principle is to continuously update a noise-free image by blending it, using a linear combination, with the raw, noisy, live image stream.

Algorithmically, this amounts to apply the following recurrence: where **a** is a mixture coefficient. The value of this coefficient can be adjusted so that a prescribed noise reduction ratio is achieved. This procedure is effective when applied to still images, but generates a trailing effect on moving objects because of the transient behavior of the filter. The larger the noise reduction ratio, the heavier the trailing effect. To work around this, a non-linearity can be introduced in the blending process: small gray-level values variations between successive images are usually caused by noise, while large variations correspond to changes in the signal itself (camera displacement or object movements). Function [EasyImage::RecursiveAverage](#) uses this observation and applies stronger noise reduction to small variations and conversely. This way, noise is better reduced in still areas and trailing is avoided in moving areas. For optimal performance, the non-linearity must be pre-computed once for all using function [EasyImage::SetRecursiveAverageLUT](#).

Note. Before the first call to the [EasyImage::RecursiveAverage](#) method, the 16-bit work image *must* be cleared (all pixel values set to zero).

EasyImage.Register

Registers an image by realigning one, two or three pivot points to reference positions.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Register(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    float sourceImagePivot0X,
    float sourceImagePivot0Y,
    float destinationImagePivot0X,
    float destinationImagePivot0Y,
    int interpolationBits
)
```

```
void Register(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,  
    float sourceImagePivot0X,  
    float sourceImagePivot0Y,  
    float destinationImagePivot0X,  
    float destinationImagePivot0Y,  
    int interpolationBits  
)
```

```
void Register(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,  
    float sourceImagePivot0X,  
    float sourceImagePivot0Y,  
    float destinationImagePivot0X,  
    float destinationImagePivot0Y,  
    int interpolationBits  
)
```

```
void Register(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,  
    float sourceImagePivot0X,  
    float sourceImagePivot0Y,  
    float sourceImagePivot1X,  
    float sourceImagePivot1Y,  
    float destinationImagePivot0X,  
    float destinationImagePivot0Y,  
    float destinationImagePivot1X,  
    float destinationImagePivot1Y,  
    int interpolationBits,  
    bool resize  
)
```

```
void Register(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,  
    float sourceImagePivot0X,  
    float sourceImagePivot0Y,  
    float sourceImagePivot1X,  
    float sourceImagePivot1Y,  
    float destinationImagePivot0X,  
    float destinationImagePivot0Y,  
    float destinationImagePivot1X,  
    float destinationImagePivot1Y,  
    int interpolationBits,  
    bool resize  
)
```

```
void Register(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,  
    float sourceImagePivot0X,  
    float sourceImagePivot0Y,  
    float sourceImagePivot1X,  
    float sourceImagePivot1Y,  
    float destinationImagePivot0X,  
    float destinationImagePivot0Y,  
    float destinationImagePivot1X,  
    float destinationImagePivot1Y,  
    int interpolationBits,  
    bool resize  
)
```

```
void Register(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,  
    float sourceImagePivot0X,  
    float sourceImagePivot0Y,  
    float sourceImagePivot1X,  
    float sourceImagePivot1Y,  
    float sourceImagePivot2X,  
    float sourceImagePivot2Y,  
    float destinationImagePivot0X,  
    float destinationImagePivot0Y,  
    float destinationImagePivot1X,  
    float destinationImagePivot1Y,  
    float destinationImagePivot2X,  
    float destinationImagePivot2Y,  
    int interpolationBits  
)
```

```
void Register(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,  
    float sourceImagePivot0X,  
    float sourceImagePivot0Y,  
    float sourceImagePivot1X,  
    float sourceImagePivot1Y,  
    float sourceImagePivot2X,  
    float sourceImagePivot2Y,  
    float destinationImagePivot0X,  
    float destinationImagePivot0Y,  
    float destinationImagePivot1X,  
    float destinationImagePivot1Y,  
    float destinationImagePivot2X,  
    float destinationImagePivot2Y,  
    int interpolationBits  
)
```

```

void Register(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    float sourceImagePivot0X,
    float sourceImagePivot0Y,
    float sourceImagePivot1X,
    float sourceImagePivot1Y,
    float sourceImagePivot2X,
    float sourceImagePivot2Y,
    float destinationImagePivot0X,
    float destinationImagePivot0Y,
    float destinationImagePivot1X,
    float destinationImagePivot1Y,
    float destinationImagePivot2X,
    float destinationImagePivot2Y,
    int interpolationBits
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. May not be the same as the source image.

sourceImagePivot0X

First pivot point abscissa in the source image.

sourceImagePivot0Y

First pivot point ordinate in the source image.

destinationImagePivot0X

First pivot point abscissa in the destination image.

destinationImagePivot0Y

First pivot point ordinate in the destination image.

interpolationBits

Number of bits of accuracy for interpolation. Allowed values are **0** (no interpolation, nearest neighbor), **4 (linear interpolation)** or **8 (cubic interpolation)**.

sourceImagePivot1X

Second pivot point abscissa in the source image.

sourceImagePivot1Y

Second pivot point ordinate in the source image.

destinationImagePivot1X

Second pivot point abscissa in the destination image.

destinationImagePivot1Y

Second pivot point ordinate in the destination image.

resize

TRUE if scaling is allowed.

sourceImagePivot2X

Third pivot point abscissa in the source image.

sourceImagePivot2Y

Third pivot point ordinate in the source image.

destinationImagePivot2X

Third pivot point abscissa in the destination image.

destinationImagePivot2Y

Third pivot point ordinate in the destination image.

Remarks

Out-of-image-bounds pixels are black.

Registration is the process of realigning two misaligned images so that point-to-point comparisons are possible. The simplest way to achieve this is to accurately locate features in both images (landmarks or pivots), using pattern matching, point measurement or whatever other technique, and realign one of the images so that the landmarks are superimposed.

* When a single pivot point is used, the registration transform is a simple translation. If interpolation bits are used, sub-pixel translation is achieved.

* When two pivot points are used, the registration is a combination of translation, rotation and optionally scaling. If scaling is not allowed, the second pivot point will not be matched exactly in general. Anyway, for most applications scaling should not be used unless it corresponds to a change of lens magnification or viewing distance.

* When three pivot points are used, the registration is a combination of translation, rotation, shearing correction and optionally scaling. The so-called shear effect can arise when acquiring images with a misaligned line-scan camera. To achieve good accuracy, the pivot points should be chosen as far apart as possible.

EasyImage.RmsNoise

Computes the root-mean-square amplitude of noise, by comparing a given image to a reference image.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
float RmsNoise(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 referenceImage,  
    Euresys.Open_eVision_2_10.EReferenceNoise referenceNoise  
    )  
  
float RmsNoise(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW16 referenceImage,  
    Euresys.Open_eVision_2_10.EReferenceNoise referenceNoise  
    )  
  
float RmsNoise(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 referenceImage,  
    Euresys.Open_eVision_2_10.EReferenceNoise referenceNoise  
    )  
  
float RmsNoise(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 referenceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    Euresys.Open_eVision_2_10.EReferenceNoise referenceNoise  
    )  
  
float RmsNoise(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW16 referenceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    Euresys.Open_eVision_2_10.EReferenceNoise referenceNoise  
    )  
  
float RmsNoise(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 referenceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    Euresys.Open_eVision_2_10.EReferenceNoise referenceNoise  
    )  
  
float RmsNoise(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW16 referenceImage,  
    uint count,  
    Euresys.Open_eVision_2_10.EReferenceNoise referenceNoise  
    )
```

Parameters

sourceImage

Pointer to the source image/ROI.

referenceImage

Pointer to the reference image/ROI.

referenceNoise

Specifies how the reference image is affected by noise, as defined by [EReferenceNoise](#).

mask

Pointer to a mask to apply the function only on a particular region in the image. Note: the mask must have the same size as the source image.

count

-

Remarks

The reference image can be noiseless (obtained by suppressing the source of noise), or affected by a noise of the same distribution as the given image.

EasyImage.ScaleRotate

Re-scales an image by an arbitrary factor and/or rotates it by an arbitrary angle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ScaleRotate(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    float sourceImagePivotX,
    float sourceImagePivotY,
    float destinationImagePivotX,
    float destinationImagePivotY,
    float scaleX,
    float scaleY,
    float rotation,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    int interpolationBits
)
```

```

void ScaleRotate(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    float sourceImagePivotX,
    float sourceImagePivotY,
    float destinationImagePivotX,
    float destinationImagePivotY,
    float scaleX,
    float scaleY,
    float rotation,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    int interpolationBits
)

void ScaleRotate(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    float sourceImagePivotX,
    float sourceImagePivotY,
    float destinationImagePivotX,
    float destinationImagePivotY,
    float scaleX,
    float scaleY,
    float rotation,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    int interpolationBits
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

sourceImagePivotX

Pivot point abscissa in the source image.

sourceImagePivotY

Pivot point ordinate in the source image.

destinationImagePivotX

Pivot point abscissa in the destination image.

destinationImagePivotY

Pivot point ordinate in the destination image.

scaleX

Scale factor for the abscissas. Its value must be different than **0.0**.

scaleY

Scale factor for the ordinates. Its value must be different than **0.0**.

rotation

Rotation angle, using the current angle unit.

destinationImage

Pointer to the destination image/ROI. May not be the same as the source image.

interpolationBits

Number of bits of accuracy for interpolation. Allowed values are **0** (no interpolation, nearest neighbor), **4 (linear interpolation)** or **8 (cubic interpolation)**.

Remarks

For resampling, the nearest neighbor rule or bilinear interpolation with 4 or 8 bits of accuracy is used.

The pivot point is a given point in the source image which is mapped to a given point in the destination image.

Rotation and scaling are done around the pivot point. The pivot point reference coordinates are based on the 'Pixel Coordinate System', meaning that the origin (0, 0) is the center of the top left pixel of the image.

Out-of-image-bounds pixels are black.

EasyImage.SetCircleWarp

Prepares suitable warp images for use with function [EasyImage::Warp](#) to unwarp a circular ring-wedge shape into a straight rectangle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetCircleWarp(
    float centerX,
    float centerY,
    int numberOfRadialSampledPoints,
    float minimumRadius,
    float maximumRadius,
    int numberOfTangentSampledPoints,
    float minimumAngle,
    float maximumAngle,
    Euresys.Open_eVision_2_10.EImageBW16 warpImageX,
    Euresys.Open_eVision_2_10.EImageBW16 warpImageY
)
```

Parameters

centerX

Abscissa of the ring-wedge center.

centerY

Ordinate of the ring-wedge center.

numberOfRadialSampledPoints

Number of points to be sampled in the radial direction.

minimumRadius

Starting radius of the ring-wedge shape.

maximumRadius

Ending radius of the ring-wedge shape.

numberOfTangentSampledPoints

Number of points to be sampled in the tangent direction.

minimumAngle

Starting angle of the ring-wedge shape.

maximumAngle

Ending angle of the ring-wedge shape.

warpImageX

Destination warp image for the abscissas.

warpImageY

Destination warp image for the ordinates.

Remarks

Typical use is unwarping of a text printed around a circle.

Note. A ring-wedge is delimited by two concentric circles and two straight lines passing through the center.

EasyImage.SetFrame

Replaces the frame of given parity in an image.

Namespace: Euresys.Open_eVision_2_10

[C#]

```

void SetFrame(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    bool odd
)

void SetFrame(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    bool odd
)

void SetFrame(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    bool odd
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. Must not be the same as the source image.

odd

Specifies which frame is replaced (the frame made up of all lines of the same parity as **odd**).

Remarks

The size of the destination image is determined as follows: **$\text{DstImage_Width} = \text{SrcImage_Width} \text{DstImage_Height} = (\text{SrcImage_Height} + 1 - \text{odd}) / 2$**

EasyImage.SetRecursiveAverageLUT

Pre-compute the required non-linear transfer function for noise reduction by recursive temporal averaging.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetRecursiveAverageLUT(
    Euresys.Open_eVision_2_10.EBW16Vector lookupTable,
    float reductionNoiseFactor,
    float reductionNoiseWidth
)
```

Parameters

lookupTable

Pointer to the LUT vector holding the non-linear transfer function.

reductionNoiseFactor

Noise reduction factor. The larger the value, the more effectively noise will be reduced.

reductionNoiseWidth

Indicates the extent to which noise reduction applies to large variations in gray-level values. For variations small with respect to this parameter, noise will be reduced by a factor close to the **reductionNoiseFactor** value; for variations much larger than **reductionNoiseWidth**, no noise reduction will take place.

Remarks

This function is a companion to [EasyImage::RecursiveAverage](#).

EasyImage.SetupEqualize

Prepares a lookup-table for image equalization, using an image histogram.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetupEqualize(
    Euresys.Open_eVision_2_10.EBWHistogramVector histogram,
    Euresys.Open_eVision_2_10.EBW8Vector lookupTable
)
```

Parameters

histogram

Pointer to the source histogram vector.

lookupTable

Pointer to the destination lookup-table vector.

Remarks

This function, along with [EasyImage::Histogram](#) and [EasyImage::Lut](#), is an alternative to using [EasyImage::Equalize](#).

EasyImage.Shrink

Resizes an image to a smaller size. Pre-filtering is applied to avoid aliasing.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Shrink(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage
)

void Shrink(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage
)

void Shrink(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI.

EasyImage.SignalNoiseRatio

Computes the signal to noise ratio, in dB, by comparing a given image to a reference image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

float SignalNoiseRatio(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 referenceImage,
    Euresys.Open_eVision_2_10.EReferenceNoise referenceNoise
)

float SignalNoiseRatio(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 referenceImage,
    Euresys.Open_eVision_2_10.EReferenceNoise referenceNoise
)

float SignalNoiseRatio(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 referenceImage,
    Euresys.Open_eVision_2_10.EReferenceNoise referenceNoise
)

float SignalNoiseRatio(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 referenceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    Euresys.Open_eVision_2_10.EReferenceNoise referenceNoise
)

float SignalNoiseRatio(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 referenceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    Euresys.Open_eVision_2_10.EReferenceNoise referenceNoise
)
```

```

float SignalNoiseRatio(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 referenceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    Euresys.Open_eVision_2_10.EReferenceNoise referenceNoise
)

float SignalNoiseRatio(
    Euresys.Open_eVision_2_10.EROIBW8 pSrcImage,
    Euresys.Open_eVision_2_10.EROIBW16 pRefImage,
    uint un32Count,
    Euresys.Open_eVision_2_10.EReferenceNoise eReferenceNoise
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

referenceImage

Pointer to the reference image/ROI.

referenceNoise

Specifies how the reference image is affected by noise, as defined by [EReferenceNoise](#).

mask

Pointer to a mask to apply the function only on a particular region in the image. Note: the mask must have the same size as the source image.

pSrcImage

-

pRefImage

-

un32Count

-

eReferenceNoise

-

Remarks

The reference image can be noiseless (obtained by suppressing the source of noise) or be affected by a noise of the same distribution as the given image.

The signal amplitude is defined as the sum of the squared pixel gray-level values while the noise amplitude is defined as the sum of the squared difference between the pixel gray-level values of the given image and the reference.

EasyImage.SwapFrames

Interchanges the even and odd rows of an image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SwapFrames (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage
)

void SwapFrames (
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage
)

void SwapFrames (
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI.

Remarks

This is helpful when acquisition of an interleaved image has confused even and odd frames.

EasyImage.Thick

Applies a thickening operation on an image, using a 3x3 kernel.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void Thick(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,  
    Euresys.Open_eVision_2_10.EKernel thickeningKernel,  
    Euresys.Open_eVision_2_10.EKernelRotation rotationMode,  
    ref int numberOfIterations  
)  
  
void Thick(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,  
    Euresys.Open_eVision_2_10.EKernel thickeningKernel,  
    Euresys.Open_eVision_2_10.EKernelRotation rotationMode,  
    ref int numberOfIterations  
)  
  
void Thick(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,  
    Euresys.Open_eVision_2_10.EKernel thickeningKernel,  
    Euresys.Open_eVision_2_10.EKernelRotation rotationMode,  
    ref int numberOfIterations  
)  
  
int Thick(  
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW1 destinationImage,  
    Euresys.Open_eVision_2_10.EKernel thickeningKernel,  
    Euresys.Open_eVision_2_10.EKernelRotation rotationMode,  
    ref int numberOfIterations  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. Must not be the same as source image.

thickeningKernel

Pointer to the thickening kernel.

rotationMode

Rotation mode, as defined by [EKernelRotation](#).

numberOfIterations

Number of iterations to apply. **0** indicates stop when convergence is reached. Upon return, gives the number of passes actually performed. If the rotation mode is set to either [Clockwise](#) or [Anticlockwise](#), a pass comprises eight kernel rotations.

Remarks

The thickening kernel coefficients must be **0** (matching black pixel, value 0), **1** (matching non black pixel, value > 0) or **-1** (don't care). When a match is found between the kernel coefficients and the neighborhood of a pixel, the pixel value is set to **255**.

EasyImage.Thin

Applies a thinning operation on an image, using a 3x3 kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void Thin(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    Euresys.Open_eVision_2_10.EKernel thinningKernel,
    Euresys.Open_eVision_2_10.EKernelRotation rotationMode,
    ref int numberOfIterations
)

void Thin(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    Euresys.Open_eVision_2_10.EKernel thinningKernel,
    Euresys.Open_eVision_2_10.EKernelRotation rotationMode,
    ref int numberOfIterations
)

void Thin(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    Euresys.Open_eVision_2_10.EKernel thinningKernel,
    Euresys.Open_eVision_2_10.EKernelRotation rotationMode,
    ref int numberOfIterations
)
```

```
int Thin(  
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW1 destinationImage,  
    Euresys.Open_eVision_2_10.EKernel thinningKernel,  
    Euresys.Open_eVision_2_10.EKernelRotation rotationMode,  
    ref int numberOfIterations  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. Must not be the same as source image.

thinningKernel

Pointer to the thinning kernel.

rotationMode

Rotation mode, as defined by [EKernelRotation](#).

numberOfIterations

Number of iterations to apply. **0** indicates stop when convergence is reached. Upon return, gives the number of passes actually performed. If the rotation mode is set to either [Clockwise](#) or [Anticlockwise](#), a pass comprises eight kernel rotations.

Remarks

The thinning kernel coefficients must be **0** (matching black pixel, value 0), **1** (matching non black pixel, value > 0) or **-1** (don't care). When a match is found between the kernel coefficients and the neighborhood of a pixel, the pixel value is set to 0.

EasyImage.ThreeLevelsMinResidueThreshold

Computes the two threshold values used to separate the pixels of an image in three classes.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
float ThreeLevelsMinResidueThreshold(  
    Euresys.Open_eVision_2_10.EBWHistogramVector histogram,  
    out Euresys.Open_eVision_2_10.EBW8 firstGrayPixelValue,  
    out Euresys.Open_eVision_2_10.EBW8 firstWhitePixelValue,  
    out float averageBlack,  
    out float averageGray,  
    out float averageWhite  
)
```

Parameters

histogram

Histogram of the image.

firstGrayPixelValue

Low threshold.

firstWhitePixelValue

High threshold.

averageBlack

Average value of the black pixels (pixels under the low threshold).

averageGray

Average value of the gray pixels (pixels between the low threshold and the high threshold).

averageWhite

Average value of the white pixels (pixels over the high threshold).

Remarks

These values are computed using the minimum residue criterion from the histogram of the image. The function returns the minimum residue as per the method. The residue is the Euclidian distance between the source image and the thresholded image.

EasyImage.Threshold

Binarize an image by setting pixels to two different possible values in a destination image, according to their value in a source image.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void Threshold(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW1 destinationImage,  
    uint threshold,  
    float relativeThreshold  
)
```

```
void Threshold(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW1 destinationImage,  
    uint threshold,  
    float relativeThreshold  
)
```

```
void Threshold(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,  
    uint threshold,  
    byte lowValue,  
    byte highValue,  
    float relativeThreshold  
)
```

```
void Threshold(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.ERRegion region,  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,  
    uint threshold,  
    byte lowValue,  
    byte highValue,  
    float relativeThreshold  
)
```

```
void Threshold(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage  
)
```

```
void Threshold(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.ERRegion region,  
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage  
)
```

```

void Threshold(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    uint threshold
)

void Threshold(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    uint threshold
)

void Threshold(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    uint threshold,
    Euresys.Open_eVision_2_10.EBW16 lowValue,
    Euresys.Open_eVision_2_10.EBW16 highValue
)

void Threshold(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    uint threshold,
    Euresys.Open_eVision_2_10.EBW16 lowValue,
    Euresys.Open_eVision_2_10.EBW16 highValue
)

void Threshold(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    float relativeThreshold
)

void Threshold(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    float relativeThreshold
)

```

```

void Threshold(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    float relativeThreshold,
    Euresys.Open_eVision_2_10.EBW16 lowValue,
    Euresys.Open_eVision_2_10.EBW16 highValue
)

void Threshold(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    float relativeThreshold,
    Euresys.Open_eVision_2_10.EBW16 lowValue,
    Euresys.Open_eVision_2_10.EBW16 highValue
)

void Threshold(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    Euresys.Open_eVision_2_10.EC24 minimum,
    Euresys.Open_eVision_2_10.EC24 maximum
)

void Threshold(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    Euresys.Open_eVision_2_10.EC24 minimum,
    Euresys.Open_eVision_2_10.EC24 maximum
)

void Threshold(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    Euresys.Open_eVision_2_10.EC24 minimum,
    Euresys.Open_eVision_2_10.EC24 maximum,
    Euresys.Open_eVision_2_10.EColorLookup colorLookupTable,
    Euresys.Open_eVision_2_10.EBW8 rejectValue,
    Euresys.Open_eVision_2_10.EBW8 acceptValue
)

```

```

void Threshold(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    Euresys.Open_eVision_2_10.EC24 minimum,
    Euresys.Open_eVision_2_10.EC24 maximum,
    Euresys.Open_eVision_2_10.EColorLookup colorLookupTable,
    Euresys.Open_eVision_2_10.EBW8 rejectValue,
    Euresys.Open_eVision_2_10.EBW8 acceptValue
)

void Threshold(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    Euresys.Open_eVision_2_10.EC24 minimum,
    Euresys.Open_eVision_2_10.EC24 maximum,
    Euresys.Open_eVision_2_10.EColorLookup colorLookupTable
)

void Threshold(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    Euresys.Open_eVision_2_10.EC24 minimum,
    Euresys.Open_eVision_2_10.EC24 maximum,
    Euresys.Open_eVision_2_10.EColorLookup colorLookupTable
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI.

threshold

The value to compare each pixel to

relativeThreshold

Fraction of the image pixels that will be set below the threshold. Only used when the threshold value is [Relative](#) (by default, **0.5**). This value must be greater than (or equal to) 0 and strictly less than 1.

lowValue

Value for pixels below the threshold (by default, **0**).

highValue

Value for pixels above the threshold (by default, it is set to **255** for BW8 destination images and **65535** for BW16 destination images).

region

Pointer to a region to apply the function only on a particular region in the image.

minimum

Three lower thresholds combined in a single color value.

maximum

Three upper thresholds combined in a single color value.

colorLookupTable

Pointer to the color lookup table to be applied before thresholding, if any.

rejectValue

Value for pixels falling outside the range (by default, **0**).

acceptValue

Value for pixels falling inside the range (by default, **255**).

Remarks

When the source image is gray-level, the pixel values are measured against a threshold. All pixels below this threshold will yield a low value in the destination image, and all pixels above or on the threshold will yield a high value. When the destination image is binary (BW1 pixel type), then the values are set to **0** or **1**, according to the criterion. When the destination image is gray-level (BW8 or BW16), then the values are set to **0** or to the maximum pixel value for the image type (**255** for BW8 and **65535** for BW16). In some overloads, these minimum and maximum destination values can be specified.

When the source image is gray-level, several modes are available: absolute (the threshold value is given), relative (the threshold value is computed to obtain a desired fraction of the image pixels), or automatic (using three different criteria). In the function overloads where this mode cannot be specified, it is assumed to be absolute.

If the source image is color, all pixels whose components are comprised in a range of values (minimum to maximum) will be set to a constant value (white by default), while other pixels will be set to another constant value (black by default). In this case, if a color lookup is specified, it is applied on the fly to the color image before thresholding.

The simpler color image overload does not support the use of an on-the-fly color lookup table nor **rejectValue/acceptValue** arguments. On the other hand, it has been MMX optimized, and will run significantly faster when the acceptance region is large.

EasyImage.TwoLevelsMinResidueThreshold

Computes the threshold value used to separate the pixels of an image in two classes.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float TwoLevelsMinResidueThreshold(
    Euresys.Open_eVision_2_10.EBWHistogramVector histogram,
    out Euresys.Open_eVision_2_10.EBW8 firstWhitePixelValue,
    out float averageBlack,
    out float averageWhite
)
```

Parameters

histogram

Histogram of the image.

firstWhitePixelValue

Threshold.

averageBlack

Average value of the black pixels (pixels under the threshold).

averageWhite

Average value of the white pixels (pixels over the threshold).

Remarks

This value is computed using the minimum residue criterion from the histogram of the image. The function returns the minimum residue as per the method. The residue is the Euclidian distance between the source image and the thresholded image.

EasyImage.Uniformize

Shading correction is the process of transforming the gray or color component values of an image, using one or two reference images or vectors.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void Uniformize(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EBW8 pixelReference,  
    Euresys.Open_eVision_2_10.EROIBW8 imageReference,  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,  
    bool multiplicative  
)  
  
void Uniformize(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EBW16 pixelReference,  
    Euresys.Open_eVision_2_10.EROIBW16 imageReference,  
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,  
    bool multiplicative  
)  
  
void Uniformize(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EC24 pixelReference,  
    Euresys.Open_eVision_2_10.EROIC24 imageReference,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,  
    bool multiplicative  
)  
  
void Uniformize(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EBW8 pixelReference,  
    Euresys.Open_eVision_2_10.EBW8Vector vectorOfPixelReference,  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,  
    bool multiplicative  
)  
  
void Uniformize(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EBW16 pixelReference,  
    Euresys.Open_eVision_2_10.EBW16Vector vectorOfPixelReference,  
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,  
    bool multiplicative  
)
```

```

void Uniformize(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EC24 pixelReference,
    Euresys.Open_eVision_2_10.EC24Vector vectorOfPixelReference,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    bool multiplicative
)

void Uniformize(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EBW8 pixelLightReference,
    Euresys.Open_eVision_2_10.EROIBW8 imageLightReference,
    Euresys.Open_eVision_2_10.EBW8 pixelDarkReference,
    Euresys.Open_eVision_2_10.EROIBW8 imageDarkReference,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage
)

void Uniformize(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EBW16 pixelLightReference,
    Euresys.Open_eVision_2_10.EROIBW16 imageLightReference,
    Euresys.Open_eVision_2_10.EBW16 pixelDarkReference,
    Euresys.Open_eVision_2_10.EROIBW16 imageDarkReference,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage
)

void Uniformize(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EC24 pixelLightReference,
    Euresys.Open_eVision_2_10.EROIC24 imageLightReference,
    Euresys.Open_eVision_2_10.EC24 pixelDarkReference,
    Euresys.Open_eVision_2_10.EROIC24 imageDarkReference,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage
)

void Uniformize(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EBW8 pixelLightReference,
    Euresys.Open_eVision_2_10.EBW8Vector vectorOfPixelLightReference,
    Euresys.Open_eVision_2_10.EBW8 pixelDarkReference,
    Euresys.Open_eVision_2_10.EBW8Vector vectorOfPixelDarkReference,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage
)

```



```

void Uniformize(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EBW16 pixelLightReference,
    Euresys.Open_eVision_2_10.EBW16Vector vectorOfPixelLightReference,
    Euresys.Open_eVision_2_10.EBW16 pixelDarkReference,
    Euresys.Open_eVision_2_10.EBW16Vector vectorOfPixelDarkReference,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage
)

void Uniformize(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EC24 pixelLightReference,
    Euresys.Open_eVision_2_10.EC24Vector vectorOfPixelLightReference,
    Euresys.Open_eVision_2_10.EC24 pixelDarkReference,
    Euresys.Open_eVision_2_10.EC24Vector vectorOfPixelDarkReference,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

pixelReference

Constant value to transform the reference image or vector into.

imageReference

Pointer to the reference source image/ROI or vector.

destinationImage

Pointer to the destination image/ROI.

multiplicative

TRUE, if the transform is multiplicative (gain); **FALSE**, if the transform is additive (offset) (by default, **TRUE**).

vectorOfPixelReference

Constant value to transform the reference image or vector into.

pixelLightReference

Constant value to transform the light reference image or vector into.

imageLightReference

Pointer to the light reference source image/ROI or vector.

pixelDarkReference

Constant value to transform the dark reference image/ROI or vector into.

imageDarkReference

Pointer to the dark reference source image/ROI or vector.

vectorOfPixelLightReference

Constant value to transform the light reference image or vector into.

vectorOfPixelDarkReference

Constant value to transform the dark reference image/ROI or vector into.

Remarks

The intent is to compensate for non-uniform lighting or sensor response non-uniformity by providing images of the background with no foreground object present.

In the case of area-scan cameras, the illumination can change anywhere in the field of view, requiring 2D compensation. In the case of line-scan cameras imaging moving parts, illumination remains constant across image rows. Only 1D compensation is required. In this case, the reference illumination is specified as a vector, which is replicated across all image rows.

* When a single reference image is used, the transform is analog to an adaptive (space-variant) gain *or* offset ($\text{Gain} * \text{Intensity}$ or $\text{Intensity} + \text{Offset}$); the transform lets the reference image(s) become a specified constant value, i.e. flat field illumination.

* When two reference images are used, the transform is analog to adaptive gain *and* offset ($\text{Gain} * \text{Intensity} + \text{Offset}$); the transform let both reference images become specified constants, i.e. flat field illumination with a correct black reference.

Note. The reference image(s) should be chosen such that they contain no saturated pixel values (remain in the linear domain) and little (filtered out) noise.

EasyImage.VerticalMirror

Mirrors an image vertically (the rows are swapped).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void VerticalMirror(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage
)

void VerticalMirror(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage
)
```

```
void VerticalMirror(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

EasyImage.Warp

Transforms an image so that each pixels are moved to the locations specified in the "warp" images used as look-up tables.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void Warp(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,  
    Euresys.Open_eVision_2_10.EImageBW16 warpImageX,  
    Euresys.Open_eVision_2_10.EImageBW16 warpImageY,  
    int shiftX,  
    int shiftY  
)  
  
void Warp(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,  
    Euresys.Open_eVision_2_10.EImageBW16 warpImageX,  
    Euresys.Open_eVision_2_10.EImageBW16 warpImageY,  
    int shiftX,  
    int shiftY  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI.

warpImageX

Pointer to the X lookup image.

warpImageY

Pointer to the Y lookup image.

shiftX

Horizontal translation.

shiftY

Vertical translation.

Remarks

For example, pixel **[10,20]** moves to location **[WarpXImage[10,20], WarpYImage[10,20]]**.

EasyImage.WeightedMoments

Computes the zero-th, first, second, third or fourth order weighted moments on the gray-level image. The weight of a pixel is its gray-level value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void WeightedMoments (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    out float M,
    out float Mx,
    out float My
)

void WeightedMoments (
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    out float M,
    out float Mx,
    out float My
)
```

```
void WeightedMoments (  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.ERegion region,  
    out float M,  
    out float Mx,  
    out float My  
)
```

```
void WeightedMoments (  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.ERegion region,  
    out float M,  
    out float Mx,  
    out float My  
)
```

```
void WeightedMoments (  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    out float M,  
    out float Mx,  
    out float My,  
    out float Mxx,  
    out float Mxy,  
    out float Myy  
)
```

```
void WeightedMoments (  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    out float M,  
    out float Mx,  
    out float My,  
    out float Mxx,  
    out float Mxy,  
    out float Myy  
)
```

```
void WeightedMoments(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.ERegion region,  
    out float M,  
    out float Mx,  
    out float My,  
    out float Mxx,  
    out float Mxy,  
    out float Myy  
)
```

```
void WeightedMoments(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.ERegion region,  
    out float M,  
    out float Mx,  
    out float My,  
    out float Mxx,  
    out float Mxy,  
    out float Myy  
)
```

```
void WeightedMoments(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    out float M,  
    out float Mx,  
    out float My,  
    out float Mxx,  
    out float Mxy,  
    out float Myy,  
    out float Mxxx,  
    out float Mxxy,  
    out float Mxyy,  
    out float Myyy  
)
```

```
void WeightedMoments(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    out float M,  
    out float Mx,  
    out float My,  
    out float Mxx,  
    out float Mxy,  
    out float Myy,  
    out float Mxxx,  
    out float Mxxy,  
    out float Mxyy,  
    out float Myyy  
)
```

```
void WeightedMoments(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.ERegion region,  
    out float M,  
    out float Mx,  
    out float My,  
    out float Mxx,  
    out float Mxy,  
    out float Myy,  
    out float Mxxx,  
    out float Mxxy,  
    out float Mxyy,  
    out float Myyy  
)
```

```
void WeightedMoments(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.ERegion region,  
    out float M,  
    out float Mx,  
    out float My,  
    out float Mxx,  
    out float Mxy,  
    out float Myy,  
    out float Mxxx,  
    out float Mxxy,  
    out float Mxyy,  
    out float Myyy  
)
```

```
void WeightedMoments (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    out float M,
    out float Mx,
    out float My,
    out float Mxx,
    out float Mxy,
    out float Myy,
    out float Mxxx,
    out float Mxxy,
    out float Mxyy,
    out float Myyy,
    out float Mxxxx,
    out float Mxxxxy,
    out float Mxxxyy,
    out float Mxyyyy,
    out float Myyyyy
)
```

```
void WeightedMoments (
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    out float M,
    out float Mx,
    out float My,
    out float Mxx,
    out float Mxy,
    out float Myy,
    out float Mxxx,
    out float Mxxy,
    out float Mxyy,
    out float Myyy,
    out float Mxxxx,
    out float Mxxxxy,
    out float Mxxxyy,
    out float Mxyyyy,
    out float Myyyyy
)
```



```
void WeightedMoments (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    out float M,
    out float Mx,
    out float My,
    out float Mxx,
    out float Mxy,
    out float Myy,
    out float Mxxx,
    out float Mxxxy,
    out float Mxyy,
    out float Myyy,
    out float Mxxxx,
    out float Mxxxxy,
    out float Mxxyy,
    out float Mxyyy,
    out float Myyyy
)
```

```
void WeightedMoments (
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.ERegion region,
    out float M,
    out float Mx,
    out float My,
    out float Mxx,
    out float Mxy,
    out float Myy,
    out float Mxxx,
    out float Mxxxy,
    out float Mxyy,
    out float Myyy,
    out float Mxxxx,
    out float Mxxxxy,
    out float Mxxyy,
    out float Mxyyy,
    out float Myyyy
)
```

```
void WeightedMoments (  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    out float M,  
    out float Mx,  
    out float My  
)
```

```
void WeightedMoments (  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    out float M,  
    out float Mx,  
    out float My  
)
```

```
void WeightedMoments (  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    out float M,  
    out float Mx,  
    out float My,  
    out float Mxx,  
    out float Mxy,  
    out float Myy  
)
```

```
void WeightedMoments (  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    out float M,  
    out float Mx,  
    out float My,  
    out float Mxx,  
    out float Mxy,  
    out float Myy  
)
```

```
void WeightedMoments(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    out float M,  
    out float Mx,  
    out float My,  
    out float Mxx,  
    out float Mxy,  
    out float Myy,  
    out float Mxxx,  
    out float Mxxy,  
    out float Mxyy,  
    out float Myyy  
)
```

```
void WeightedMoments(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 mask,  
    out float M,  
    out float Mx,  
    out float My,  
    out float Mxx,  
    out float Mxy,  
    out float Myy,  
    out float Mxxx,  
    out float Mxxy,  
    out float Mxyy,  
    out float Myyy  
)
```

```

void WeightedMoments (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    out float M,
    out float Mx,
    out float My,
    out float Mxx,
    out float Mxy,
    out float Myy,
    out float Mxxx,
    out float Mxxy,
    out float Mxyy,
    out float Myyy,
    out float Mxxxx,
    out float Mxxxxy,
    out float Mxxxyy,
    out float Mxyyy,
    out float Myyyy
)

void WeightedMoments (
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 mask,
    out float M,
    out float Mx,
    out float My,
    out float Mxx,
    out float Mxy,
    out float Myy,
    out float Mxxx,
    out float Mxxy,
    out float Mxyy,
    out float Myyy,
    out float Mxxxx,
    out float Mxxxxy,
    out float Mxxxyy,
    out float Mxyyy,
    out float Myyyy
)

```

Parameters

sourceImage

Pointer to the source image/ROI.

M

Reference to the zero-th order weighted moment (total gray value).

M_x

Reference to the first order moments (weighted sums of abscissas and ordinates).

M_y

Reference to the first order moments (weighted sums of abscissas and ordinates).

region

Pointer to a region to apply the function only on a particular region in the image.

M_{xx}

Reference to the second order uncentered moments (weighted sums of squared abscissas, cross-product and squared ordinates).

M_{xy}

Reference to the second order uncentered moments (weighted sums of squared abscissas, cross-product and squared ordinates).

M_{yy}

Reference to the second order uncentered moments (weighted sums of squared abscissas, cross-product and squared ordinates).

M_{xxx}

Reference to the third order uncentered moments (weighted sums of third order products).

M_{xxxy}

Reference to the third order uncentered moments (weighted sums of third order products).

M_{xxyy}

Reference to the third order uncentered moments (weighted sums of third order products).

M_{yyyy}

Reference to the third order uncentered moments (weighted sums of third order products).

M_{xxxx}

Reference to the fourth order uncentered moments (weighted sums of fourth order products).

M_{xxxxy}

Reference to the fourth order uncentered moments (weighted sums of fourth order products).

M_{xxxyy}

Reference to the fourth order uncentered moments (weighted sums of fourth order products).

M_{xyyyy}

Reference to the fourth order uncentered moments (weighted sums of fourth order products).

M_{yyyyy}

Reference to the fourth order uncentered moments (weighted sums of fourth order products).

mask

Pointer to a mask to apply the function only on a particular region in the image. Note: the mask must have the same size as the source image.

EasyImage.WhiteTopHatBox

Performs a top-hat filtering on an image (source image minus opened image) on a rectangular kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void WhiteTopHatBox(
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW1 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)

void WhiteTopHatBox(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)

void WhiteTopHatBox(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)

void WhiteTopHatBox(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    uint halfOfKernelWidth,
    uint halfOfKernelHeight
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. Must not be the same as the source image.

halfOfKernelWidth

Half of the box width minus one (by default, **halfOfKernelWidth = 1**; **0** is allowed).

halfOfKernelHeight

Half of the box height minus one (by default, same as **halfOfKernelWidth**; **0** is allowed).

Remarks

This filter enhances the thin white features.

EasyImage.WhiteTopHatDisk

Performs a top-hat filtering on an image (source image minus opened image) on a quasi-circular kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void WhiteTopHatDisk(
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW1 destinationImage,
    uint halfOfKernelWidth
)

void WhiteTopHatDisk(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    uint halfOfKernelWidth
)

void WhiteTopHatDisk(
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW16 destinationImage,
    uint halfOfKernelWidth
)

void WhiteTopHatDisk(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    uint halfOfKernelWidth
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination image/ROI. Must not be the same as the source image.

halfOfKernelWidth

Half width of the kernel minus one (by default, **halfOfKernelWidth = 1**; **0** is allowed).

Remarks

This filter enhances the thin white features.

4.15. EasyObject Class

This class contains static properties and methods specific to the EasyObject library.

Namespace: Euresys.Open_eVision_2_10

Methods

[ContourArea](#)

Computes the area of an object from its contour.

[ContourGravityCenter](#)

Computes the area and gravity center of an object from its contour.

[ContourInertia](#)

Computes the inertia parameters of an object from its contour.

[IsFloatFeature](#)

Tests whether a given feature is associated with floating-point values.

[IsIntegerFeature](#)

Tests whether a given feature is associated with integer values.

IsUnsignedIntegerFeature

Tests whether a given feature is associated with unsigned integer values.

syObject.ContourArea

Computes the area of an object from its contour.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void ContourArea (  
    Euresys.Open_eVision_2_10.EPathVector pPathVector,  
    ref int n32Area  
)
```

Parameters

pPathVector

Pointer to the source vector.

n32Area

Reference to the area to compute.

EasyObject.ContourGravityCenter

Computes the area and gravity center of an object from its contour.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void ContourGravityCenter(  
    Euresys.Open_eVision_2_10.EPathVector pPathVector,  
    ref int n32Area,  
    ref float f32GravityCenterX,  
    ref float f32GravityCenterY  
)
```

Parameters

pPathVector

Pointer to the source vector.

n32Area

Reference to the area to compute.

f32GravityCenterX

Reference to the abscissa of the gravity center to compute.

f32GravityCenterY

Reference to the ordinate of the gravity center to compute.

EasyObject.ContourInertia

Computes the inertia parameters of an object from its contour.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void ContourInertia(  
    Euresys.Open_eVision_2_10.EPathVector pPathVector,  
    ref int n32Area,  
    ref float f32GravityCenterX,  
    ref float f32GravityCenterY,  
    ref float f32SigmaX,  
    ref float f32SigmaY,  
    ref float f32SigmaXY  
)
```

Parameters

pPathVector

Pointer to the source vector.

n32Area

Reference to the area to compute.

f32GravityCenterX

Reference to the abscissa of the gravity center to compute.

f32GravityCenterY

Reference to the ordinate of the gravity center to compute.

f32SigmaX

Centered cross moment of inertia.

f32SigmaY

Centered moment of inertia around Y.

f32SigmaXY

Centered cross moment of inertia.

EasyObject.IsFloatFeature

Tests whether a given feature is associated with floating-point values.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool IsFloatFeature(
    Euresys.Open_eVision_2_10.EFeature feature
)
```

Parameters

feature

The feature.

Remarks

Most features are floating-point. The exceptions are listed in these functions: [EasyObject::IsUnsignedIntegerFeature](#) and [EasyObject::IsIntegerFeature](#).

EasyObject.IsIntegerFeature

Tests whether a given feature is associated with integer values.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool IsIntegerFeature (
    Euresys.Open_eVision_2_10.EFeature feature
)
```

Parameters

feature

The feature.

Remarks

The features associated with integer values are: [ContourX](#), [ContourY](#), [LeftLimit](#), [RightLimit](#), [TopLimit](#), and [BottomLimit](#).

EasyObject.IsUnsignedIntegerFeature

Tests whether a given feature is associated with unsigned integer values.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool IsUnsignedIntegerFeature (
    Euresys.Open_eVision_2_10.EFeature feature
)
```

Parameters

feature

The feature.

Remarks

The features associated with unsigned integer values are: [ElementIndex](#), [LayerIndex](#), [RunCount](#), [Area](#) and [LargestRun](#).

4.16. EBarcode Class

Manages a complete context for the reading or verification of bar codes in EasyBarcode.

Base Class: [ERectangleShape](#)

Namespace: Euresys.Open_eVision_2_10

Properties

[AdditionalSymbologies](#)

Enabled symbologies belonging to the group of additional symbologies.

[KnownLocation](#)

Flag indicating whether the symbol location is known or not.

[KnownModule](#)

Flag indicating whether the symbol module is known or not.

[Module](#)

Module value.

[NumDecodedSymbologies](#)

Number of symbologies (among the enabled ones) for which the decoding process was successful.

[NumEnabledSymbologies](#)

Number of enabled symbologies.

Rectangle

Sets the nominal position (center coordinates), size and rotation angle of the symbol bounding box according to a known rectangle.

RelativeReadingSizeX

Reading area width, relative to the symbol extent.

RelativeReadingSizeY

Reading area height, relative to the symbol extent.

RelativeReadingX

Reading area abscissa, relative to the symbol position.

RelativeReadingY

Reading area ordinate, relative to the symbol position.

StandardSymbologies

Enabled symbologies belonging to the group of standard symbologies.

ThicknessRatio

Bars thickness ratio.

VerifyChecksum

M "VerifyChecksum" mode.

e

thods

Decode

Provides the decoded information (or a reading error code) corresponding to the specified symbology.

Detect

Processes the image, reads the possible contents of the bar code corresponding to all enabled symbologies and rates their likelihood.

Drag

Moves a handle to a new position and updates the position parameters of the symbol bounding box.

Draw

Draws the symbol bounding box.

DrawWithCurrentPen

Draws the symbol bounding box.

EBarCode

Creates an [EBarCode](#) object.

GetDecodedAngle

Allows retrieving the bar code reading angle, pertaining to the specified symbology, or, at least, to the most likely symbology.

GetDecodedDirection

Returns the encoding direction of the bar code, pertaining to the specified symbology, or, at least, to the most likely symbology.

GetDecodedRectangle

Allows retrieving the bar code reading rectangle, pertaining to the specified symbology, or, at least, to the most likely symbology.

GetDecodedSymbology

Returns the identifier of one of the symbologies that were successfully decoded.

GetSymbologyName

Retrieves the name of given symbology as a string.

HitTest

Checks whether the cursor is positioned over a handle (**TRUE**) or not (**FALSE**).

Read

Processes the image, reads the possible contents of the bar code and returns the single possible decoding (mono-symbology) or the most likely one (multi-symbology) or a reading error code.

SetReadingCenter

Sets the reading area center coordinates, relative to the symbol position and extent.

SetReadingSize

☐ Sets the reading area size, relative to the symbol extent.

B

arCode.AdditionalSymbologies

Enabled symbologies belonging to the group of additional symbologies.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint AdditionalSymbologies
{ get; set; }
```


Remarks

Due to the large number of supported symbologies, they have been gathered in two groups. For more information about these groups, see [ESymbologies](#).

EBarcode.Decode

Provides the decoded information (or a reading error code) corresponding to the specified symbology.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
string Decode(
    Euresys.Open_eVision_2_10.ESymbologies symbology
)
```

Parameters

symbology

Specified symbology, as defined by [ESymbologies](#) this symbology must have been enabled).

Remarks

Before calling [EBarcode::Decode](#), an [EBarcode::Detect](#) operation must have been performed. In case of the mono-symbology mode, or if only the most likely decoding matters, the [EBarcode::Read](#) method should be used.

EBarcode.Detect

Processes the image, reads the possible contents of the bar code corresponding to all enabled symbologies and rates their likeliness.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void Detect(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage  
)
```

Parameters

sourceImage

Pointer to the image containing the bar code.

Remarks

Processing the image means finding the symbol (in case of automatic location mode) and retrieving its descriptive parameters. The decoded information corresponding to a specific symbology is provided by the decode function. The symbologies that were successfully decoded are ranked by decreasing likeliness (range **0** to **NumDecodedSymbologies-1**). In case of the mono-symbology mode or if only the most likely decoding matters, the **Read** function should be used.

EBarCode.Drag

Moves a handle to a new position and updates the position parameters of the symbol bounding box.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void Drag(  
    int cursorX,  
    int cursorY  
)
```

Parameters

cursorX

Cursor current coordinates.

cursorY

Cursor current coordinates.

EBarcode.Draw

Draws the symbol bounding box.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)

void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingMode

Indicates how the symbol bounding box must be displayed, as defined by [EDrawingMode](#).

daughters

TRUE if the shapes attached to the symbol bounding box are to be displayed as well.

color

The color in which to draw the overlay.

Remarks

The bounding box corresponds to the nominal position of the bar code ([Nominal](#)), in case this information has been explicitly provided, and to the actual position ([Actual](#)) if it has been determined by image analysis.

EBarcode.DrawWithCurrentPen

Draws the symbol bounding box.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawWithCurrentPen(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingMode

Indicates how the symbol bounding box must be displayed, as defined by [EDrawingMode](#).

daughters

TRUE if the shapes attached to the symbol bounding box are to be displayed as well.

Remarks

The bounding box corresponds to the nominal position of the bar code ([Nominal](#)), in case this information has been explicitly provided, and to the actual position ([Actual](#)) if it has been determined by image analysis.

EBarcode.EBarcode

Creates an [EBarcode](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EBarcode(
)

void EBarcode(
    Euresys.Open_eVision_2_10.EBarcode other
)
```

Parameters

other

-

EBarcode.GetDecodedAngle

Allows retrieving the bar code reading angle, pertaining to the specified symbology, or, at least, to the most likely symbology.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetDecodedAngle(
    out float decodedAngle
)

void GetDecodedAngle(
    out float decodedAngle,
    float cutAngle
)

void GetDecodedAngle(
    Euresys.Open_eVision_2_10.ESymbologies symbology,
    out float decodedAngle
)
```

```
void GetDecodedAngle(  
    Euresys.Open_eVision_2_10.ESymbologies symbology,  
    out float decodedAngle,  
    float cutAngle  
)
```

Parameters

decodedAngle

Returned bar code reading angle value.

cutAngle

Cut angle value (°) defining the allowed range for the bar code reading angle (**[cutAngle, cutAngle + 360]**). By default, the cut angle equals **-45**.

symbology

Specified symbology, as defined by [ESymbologies](#) (this symbology must have been enabled).

EBarCode.GetDecodedDirection

Returns the encoding direction of the bar code, pertaining to the specified symbology, or, at least, to the most likely symbology.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void GetDecodedDirection(  
    out bool directEncoding  
)  
  
void GetDecodedDirection(  
    Euresys.Open_eVision_2_10.ESymbologies symbology,  
    out bool directEncoding  
)
```

Parameters

directEncoding

Boolean holding the encoding direction status.

symbology

Specified symbology, as defined by [ESymbologies](#) (this symbology must have been enabled).

Remarks

The encoding direction of the bar code is "Direct" or "Inverse". The encoding direction is said to be "Direct" when the bar code longitudinal axis falls in the range [-45°, 135]. Conversely, when the bar code longitudinal axis doesn't fall in the previous range, the encoding direction is said to be "Inverse".

EBarcode.GetDecodedRectangle

Allows retrieving the bar code reading rectangle, pertaining to the specified symbology, or, at least, to the most likely symbology.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetDecodedRectangle(
    Euresys.Open_eVision_2_10.ERectangle rect
)

void GetDecodedRectangle(
    Euresys.Open_eVision_2_10.ESymbologies symbology,
    Euresys.Open_eVision_2_10.ERectangle rect
)
```

Parameters

rect

Returned bar code reading rectangle.

symbology

Specified symbology, as defined by [ESymbologies](#) (this symbology must have been enabled).

EBarcode.GetDecodedSymbology

Returns the identifier of one of the symbologies that were successfully decoded.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ESymbologies GetDecodedSymbology(
    uint index
)
```

Parameters

index

Index of the specified symbology (range **0** to **NumDecodedSymbologies-1**).

Remarks

The desired symbology is specified by its ranking index (range **0** to **NumDecodedSymbologies-1**). The symbologies that were successfully decoded are ranked by decreasing likeliness.

EBarcode.GetSymbologyName

Retrieves the name of given symbology as a string.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
string GetSymbologyName(
    Euresys.Open_eVision_2_10.ESymbologies symbology
)
```

Parameters

symbology

Symbology.

EBarcode.HitTest

Checks whether the cursor is positioned over a handle (**TRUE**) or not (**FALSE**).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
bool HitTest(  
    bool daughters  
)
```

Parameters

daughters

TRUE if the handles of the shapes attached to the symbol bounding box have to be considered as well.

EBarcode.KnownLocation

Flag indicating whether the symbol location is known or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
bool KnownLocation  
  
    { get; set; }
```

Remarks

In case of known location, use [EBarcode::Rectangle](#) to adjust a rectangle around the symbol.

EBarcode.KnownModule

Flag indicating whether the symbol module is known or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool KnownModule
{ get; set; }
```

Remarks

If **TRUE**, it is also necessary to get the [EBarcode::Module](#) and [EBarcode::ThicknessRatio](#) properties in order to specify the requested module and thickness ratio.

EBarcode.Module

Module value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float Module
{ get; set; }
```

Remarks

The module value is a descriptive parameter participating in the encoding; it corresponds to the thinner bar width. Symbols whose bars thickness can take two values, the module and **V** times the module (where **V** runs from **1.5** to **3**), are called *binary* bar codes. When the bars thickness are small integer multiples (1 to 4 or 5) of a module, the symbols are called *modular* bar codes.

EBarcode.NumDecodedSymbologies

Number of symbologies (among the enabled ones) for which the decoding process was successful.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint NumDecodedSymbologies
{ get; }
```

EBarcode.NumEnabledSymbologies

Number of enabled symbologies.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint NumEnabledSymbologies
{ get; }
```

EBarcode.Read

Processes the image, reads the possible contents of the bar code and returns the single possible decoding (mono-symbology) or the most likely one (multi-symbology) or a reading error code.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
string Read(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage
)
```

Parameters

sourceImage

The image containing the bar code.

Remarks

Processing the image means finding the symbol (in case of automatic location mode) and retrieving its descriptive parameters. When decoding other than the most likely one are also required, a call to the [EBarcode::Detect](#) function followed by a [EBarcode::Decode](#) should be used.

EBarcode.Rectangle

Sets the nominal position (center coordinates), size and rotation angle of the symbol bounding box according to a known rectangle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
override Euresys.Open_eVision_2_10.ERectangle Rectangle
    { get; set; }
```

Remarks

An [ERectangle](#) object is characterized by its center coordinates, its size and its rotation angle.

EBarcode.RelativeReadingSizeX

Reading area width, relative to the symbol extent.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float RelativeReadingSizeX
    { get; }
```

EBarcode.RelativeReadingSizeY

Reading area height, relative to the symbol extent.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float RelativeReadingSizeY  
    { get; }
```

EBarcode.RelativeReadingX

Reading area abscissa, relative to the symbol position.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float RelativeReadingX  
    { get; }
```

EBarcode.RelativeReadingY

Reading area ordinate, relative to the symbol position.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float RelativeReadingY
{ get; }
```

EBarcode.SetReadingCenter

Sets the reading area center coordinates, relative to the symbol position and extent.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetReadingCenter(
    float relativeX,
    float relativeY
)
```

Parameters

relativeX

Reading area center abscissa, relative to the symbol position and extent.

relativeY

Reading area center ordinate, relative to the symbol position and extent.

EBarcode.SetReadingSize

Sets the reading area size, relative to the symbol extent.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetReadingSize(
    float relativeSizeX,
    float relativeSizeY
)
```

Parameters

relativeSizeX

Reading area width, relative to the symbol extent.

relativeSizeY

Reading area height, relative to the symbol extent.

EBarcode.StandardSymbologies

Enabled symbologies belonging to the group of standard symbologies.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint StandardSymbologies
{ get; set; }
```

Remarks

Due to the large number of supported symbologies, they have been gathered in two groups. For more information about these groups, see [ESymbologies](#).

EBarcode.ThicknessRatio

Bars thickness ratio.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ThicknessRatio
```

```
{ get; set; }
```

Remarks

This property is relevant in case of binary codes only. It corresponds to the ratio of a thin bar width over a thick bar width, and should range from **1.5** to **3**.

EBarcode.VerifyChecksum

"VerifyChecksum" mode.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
bool VerifyChecksum
```

```
{ get; set; }
```

Remarks

The "VerifyChecksum" mode enables or disables verification of the checksum character. That verification mode is set in the same way for all enabled symbologies. It is worth noting that checksum may be present or not in the bar code, and the user may verify or not checksum validity. These two circumstances are independent, and give rise to four modes of operation. The verification process will return an "invalid checksum" error in case of bad checksum character. This error can also be generated if there is no checksum in the bar code. In the other case, when checksum are not verified, no error will occur, and the process will continue silently. When the "VerifyChecksum" mode is enabled, the returned decoded string does not contain the checksum character(s). Conversely, when the verification process is disabled, the checksum character(s) are concatenated to the encoded information.

4.17. EBaseROI Class

This represents the abstract base class for all ROI and image classes.

Derived Class(es): [EROIC24](#) [EROIBW8](#) [EROIBW16](#) [EROIBW32](#) [EROIC15](#) [EROIC16](#) [EROIC24A](#) [EROIBW1](#) [EROIC48](#)

Namespace: Euresys.Open_eVision_2_10

Properties

Author

Gets or sets the Author attribute of the ROI/image. This is a convenience property that allows storing some information in the ROI/image. Please note that not all file formats preserve this information.

BaseTopParent

Returns the image at the top of the hierarchy, or NULL if there is no image on top.

BitsPerPixel

Gets the number of storage bits per pixel.

ColorSystem

Gets or sets the color system used by this image, as defined by the [EColorSystem](#) enumeration.

ColPitch

Gets the pitch of a column (number of bytes between two horizontally adjacent pixels). For BW1 (one bit per pixel) images, this value is undefined.

Comment

Gets or sets the Comment attribute of the ROI/image. This is a convenience property that allows storing some information in the ROI/image. Please note that not all file formats preserve this information.

Date

Gets or sets the Date attribute of the ROI/image. This is a convenience property that allows storing some information in the ROI/image. Please note that not all file formats preserve this information.

Height

Gets or sets the height of the ROI.

IsVoid

Tests whether if the topmost parent image of this hierarchy has a zero size.

OrgX

Gets or sets the x-coordinate of the upper left corner of the ROI, relative to its parent ROI/image.

OrgY

Gets or sets the y-coordinate of the upper left corner of the ROI, relative to its parent ROI/image.

Parent

Returns the hierarchical parent of this object.

PlanesPerPixel

Gets the number of color components in each pixel of the ROI/image.

RowPitch

Gets the pitch of a row (the number of bytes between two vertically adjacent pixels). This is also valid for BW1 images (one bit per pixel).

Title

Gets or sets the Title attribute of the ROI/image. This is a convenience property that allows storing some information in the ROI/image. Please note that not all file formats preserve this information.

TotalHeight

Gets the height, in pixels, of the ROI topmost parent.

TotalOrgX

Gets the x-coordinate of the upper left pixel of the ROI, relative to its topmost parent upper left pixel.

TotalOrgY

Gets the y-coordinate of the upper left pixel of the ROI, relative to its topmost parent upper left pixel.

TotalWidth

Gets the width, in pixels, of the ROI topmost parent.

Type

Gets the ROI/image pixel type, as defined by the [EImageType](#) enumeration.

Width

M Gets or sets the width of the ROI.

e

thods

Attach

This method attaches the ROI to another ROI or image. Only ROIs can be attached. An image can never be attached to another image or ROI. Optionally, the ROI can be moved and resized after attachment by supplying additional parameters.

CopyTo

Copies all the data of the current [EBaseROI](#) object into another [EBaseROI](#) object and returns it.

CropToImage

This method reduces the ROI size so that it is completely contained within its topmost parent image bounds (if such an image exists at the top of the hierarchy). This means that, after calling this method, either the ROI has a zero size, or all of its pixels map to valid pixels of the underlying image buffer. If the ROI is already contained within its parent image bounds, then this method does nothing.

Drag

Moves the specified handle to a new position and updates all placement parameters of the ROI.

Draw

Draws an ROI/image in a device context.

DrawFrame

Draws a rectangular frame around an image or ROI.

DrawFrameWithCurrentPen

Draws a rectangular frame around an image or ROI.

GetImagePtr

Returns a pointer to the pixel at given coordinates within the image/ROI.

GetSubBaseROIs

Returns all the children, and possibly recursively all their children, too.

HasSubROI

Tests whether this object has a given attached ROI.

HitTest

Detects if the cursor is placed over one of the dragging handles.

IsAnROI

Tests whether this object is an ROI or an image.

Load

Restores an image stored in the given file.

Save

Saves the [EBaseROI](#) object to the given file.

SaveJpeg

Saves the [EBaseROI](#) object to the given file, in JPEG format.

SaveJpeg2K

Saves the [EBaseROI](#) object to the given file, in JPEG 2000 format.

SavePng

Saves the [EBaseROI](#) object to the given file, in PNG format.

Serialize

Serializes the [EBaseROI](#).

SetImagePtr

Sets the pointer to an externally allocated image buffer.

SetPlacement

Sets the placement of an ROI, relative to its parent ROI/image.

SetSize

Sets the width and height of an ROI/image.

B

BaseROI.Attach

This method attaches the ROI to another ROI or image.
Only ROIs can be attached. An image can never be attached to another image or ROI.
Optionally, the ROI can be moved and resized after attachment by supplying additional parameters.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Attach(
    Euresys.Open_eVision_2_10.EBaseROI parent
)

void Attach(
    Euresys.Open_eVision_2_10.EBaseROI parent,
    int orgX,
    int orgY,
    int width,
    int height
)
```

Parameters

parent

ROI or image on which to attach the ROI.

orgX

When specified, sets the new x-coordinate of the ROI top-left corner.

orgY

When specified, sets the new y-coordinate of the ROI top-left corner.

width

When specified, sets the new width of the ROI.

height

When specified, sets the new height of the ROI.

EBaseROI.Author

Gets or sets the Author attribute of the ROI/image. This is a convenience property that allows storing some information in the ROI/image. Please note that not all file formats preserve this information.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
string Author  
    { get; set; }
```

EBaseROI.BaseTopParent

Returns the image at the top of the hierarchy, or NULL if there is no image on top.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EBaseROI BaseTopParent  
    { get; }
```

EBaseROI.BitsPerPixel

Gets the number of storage bits per pixel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint BitsPerPixel  
  
    { get; }
```

EBaseROI.ColorSystem

Gets or sets the color system used by this image, as defined by the [EColorSystem](#) enumeration.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EColorSystem ColorSystem  
  
    { get; set; }
```

Remarks

Upon object creation, a default color system is set, compatible with the ROI/image type ([GrayLevel](#) for gray-level types and [Rgb](#) for color types).

The color system associated to an image is mainly relevant when working on color images. See [EasyColor](#) (FG) for more information.

EBaseROI.ColPitch

Gets the pitch of a column (number of bytes between two horizontally adjacent pixels). For BW1 (one bit per pixel) images, this value is undefined.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
int ColPitch
{ get; }
```

EBaseROI.Comment

Gets or sets the Comment attribute of the ROI/image. This is a convenience property that allows storing some information in the ROI/image. Please note that not all file formats preserve this information.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
string Comment
{ get; set; }
```

EBaseROI.CopyTo

Copies all the data of the current [EBaseROI](#) object into another [EBaseROI](#) object and returns it.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void CopyTo(
    Euresys.Open_eVision_2_10.EBaseROI dest
)
```

Parameters

dest

Pointer to the [EBaseROI](#) object in which the current [EBaseROI](#) object data have to be copied.

Remarks

This method copies all the object data to the destination object. The attached ROIs are copied recursively and attached to the destination object. They will be deleted automatically when the destination object is deleted.

When the buffer of the source image has been provided by a call to **SetImagePtr**, the pointer will be copied into the destination image. Both images will thus refer the same external buffer.

EBaseROI.CropToImage

This method reduces the ROI size so that it is completely contained within its topmost parent image bounds (if such an image exists at the top of the hierarchy). This means that, after calling this method, either the ROI has a zero size, or all of its pixels map to valid pixels of the underlying image buffer. If the ROI is already contained within its parent image bounds, then this method does nothing.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void CropToImage (
)
```

EBaseROI.Date

Gets or sets the Date attribute of the ROI/image. This is a convenience property that allows storing some information in the ROI/image. Please note that not all file formats preserve this information.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
string Date
{ get; set; }
```

EBaseROI.Drag

Moves the specified handle to a new position and updates all placement parameters of the ROI.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Drag(
    Euresys.Open_eVision_2_10.EDragHandle dragHandle,
    int x,
    int y,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

Parameters

dragHandle

Handle identifier, as defined by [EDragHandle](#). The value returned by [EBaseROI::HitTest](#) should be used.

x

x-coordinate of the mouse cursor.

y

y-coordinate of the mouse cursor.

zoomX

Horizontal zoom factor. By default, true scale is used.

zoomY

Vertical zoom factor. By default, true scale is used.

panX

Horizontal pan offset. By default, no pan is added.

panY

Vertical pan offset. By default, no pan is added.

Remarks

If zooming and/or panning were used when drawing the ROI, the same values must be used with [EBaseROI::HitTest](#) and [EBaseROI::Drag](#).

EBaseROI.Draw

Draws an ROI/image in a device context.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EBw8Vector bw8Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

```

Parameters

graphicContext

Handle to the device context of the destination window.

zoomX

Magnification factor for zooming in or out in the horizontal direction. By default, the image is displayed in **TRUE** scale.

zoomY

Magnification factor for zooming in or out in the vertical direction. Setting a **0** value (which is the default) will result in isotropic scaling (i.e. equal horizontal and vertical factors).

panX

Pan offset (in pixels) in the horizontal direction. By default, no panning is applied.

panY

Pan offset (in pixels) in the vertical direction. By default, no panning is applied.

c24Vector

When supplied, this parameter allows using a LUT that maps from BW8 to C24 when drawing (false colors).

bw8Vector

When supplied, this parameter allows using a LUT that maps from BW8 to BW8 when drawing.

Remarks

An ROI/image can be drawn (its pixels rendered) using a device context.

The horizontal and vertical zooming factors can be different and must be contained in the **1/16..16** range.

(MFC users can use the **CDC::GetSafeHdc()** method to obtain a suitable device context handle from a **CDC** instance.)

EBaseROI.DrawFrame

Draws a rectangular frame around an image or ROI.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void DrawFrame(  
    IntPtr graphicContext,  
    bool handles,  
    float zoomX,  
    float zoomY,  
    float x,  
    float y  
)  
  
void DrawFrame(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EFramePosition framePosition,  
    bool handles,  
    float zoomX,  
    float zoomY,  
    float x,  
    float y  
)  
  
void DrawFrame(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ERGBColor color,  
    bool handles,  
    float zoomX,  
    float zoomY,  
    float x,  
    float y  
)
```

```

void DrawFrame (
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.EFramePosition framePosition,
    bool handles,
    float zoomX,
    float zoomY,
    float x,
    float y
)

void DrawFrame (
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    bool handles,
    float zoomX,
    float zoomY,
    float x,
    float y
)

```

Parameters

graphicContext

Handle to the device context of the destination window.

handles

TRUE if handles are to be drawn.

zoomX

Magnification factor for zooming in or out in the horizontal direction. By default, the image is displayed in **TRUE** scale.

zoomY

Magnification factor for zooming in or out in the vertical direction. Setting a **0** value (which is the default) will result in isotropic scaling (i.e. equal horizontal and vertical factors).

x

Translation factor for panning in the horizontal direction.

y

Translation factor for panning in the vertical direction.

framePosition

Positioning of the frame relative to the ROI.

color

Color in which to draw the frame.

Remarks

A frame can be drawn (using a device context) around an image or ROI, possibly with 9 sizing handles.

A suitable default pen is used (see [EBaseROI::DrawFrameWithCurrentPen](#) if you wish to use the pen currently selected into the device context).

Zooming and panning are possible. Please note that panning is applied *before* zooming.

(MFC users can use the **CDC::GetSafeHdc()** method to obtain a suitable device context handle from a **CDC** instance.)

EBaseROI.DrawFrameWithCurrentPen

Draws a rectangular frame around an image or ROI.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void DrawFrameWithCurrentPen (
    IntPtr graphicContext,
    bool handles,
    float zoomX,
    float zoomY,
    float x,
    float y
)

void DrawFrameWithCurrentPen (
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EFramePosition framePosition,
    bool handles,
    float zoomX,
    float zoomY,
    float x,
    float y
)
```

Parameters

graphicContext

Handle to the device context of the destination window.

handles

TRUE if handles are to be drawn.

zoomX

Magnification factor for zooming in or out in the horizontal direction. By default, the image is displayed in **TRUE** scale.

zoomY

Magnification factor for zooming in or out in the vertical direction. Setting a **0** value (which is the default) will result in isotropic scaling (i.e. equal horizontal and vertical factors).

x

Translation factor for panning in the horizontal direction.

y

Translation factor for panning in the vertical direction.

framePosition

Positioning of the frame relative to the ROI.

Remarks

A frame can be drawn (using a device context) around an image or ROI, possibly with 9 sizing handles.

The current device context pen is used. Zooming and panning are possible. Please note that panning is applied *before* zooming.

(MFC users can use the **CDC::GetSafeHdc()** method to obtain a suitable device context handle from a **CDC** instance.)

EBaseROI.GetImagePtr

Returns a pointer to the pixel at given coordinates within the image/ROI.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
IntPtr GetImagePtr (
    int x,
    int y
)

IntPtr GetImagePtr (
    int x,
    int y
)

IntPtr GetImagePtr (
)
```

```
IntPtr GetImagePtr(  
    )
```

Parameters

x

The pixel x-coordinate.

y

The pixel y-coordinate.

Remarks

This methods returns the memory address of the byte that contains the pixel (or address that contains the first byte of the pixel if it is bigger than one byte).

If the pixel coordinates are not specified, the method returns the address of the top-left pixel of the ROI/image.

EBaseROI.GetSubBaseROIs

Returns all the children, and possibly recursively all their children, too.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EBaseROI[] GetSubBaseROIs(  
    bool recursive  
    )  
Euresys.Open_eVision_2_10.EBaseROI[] GetSubBaseROIs(  
    bool recursive  
    )
```

Parameters

recursive

TRUE to retrieve all sub-ROIs recursively. FALSE otherwise.

EBaseROI.HasSubROI

Tests whether this object has a given attached ROI.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool HasSubROI (
    Euresys.Open_eVision_2_10.EBaseROI subROI
)
```

Parameters

subROI

Sub ROI to find.

EBaseROI.Height

Gets or sets the height of the ROI.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int Height
{ get; set; }
```

Remarks

The *placement of an ROI* is given by the coordinates its upper left corner (origin point), relative to its parent ROI/image and by its size (width and height).

EBaseROI.HitTest

Detects if the cursor is placed over one of the dragging handles.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EDragHandle HitTest(
    int x,
    int y,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

Parameters

x

x-coordinate of the mouse cursor.

y

y-coordinate of the mouse cursor.

zoomX

Horizontal zoom factor. By default, true scale is used.

zoomY

Vertical zoom factor. By default, true scale is used.

panX

Horizontal pan offset. By default, no pan is added.

panY

Vertical pan offset. By default, no pan is added.

Remarks

Returns a handle identifier, as defined by [EDragHandle](#).

If zooming and/or panning were used when drawing the ROI, the same values must be used with [EBaseROI::HitTest](#) and [EBaseROI::Drag](#).

EBaseROI.IsAnROI

Tests whether this object is an ROI or an image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool IsAnROI (  
    )
```

EBaseROI.IsVoid

Tests whether if the topmost parent image of this hierarchy has a zero size.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool IsVoid  
    { get; }
```

Remarks

For an image, this method returns **TRUE** if the image size is zero.

For an ROI, this method returns **TRUE** if the topmost parent image size is zero or if there is no topmost image.

EBaseROI.Load

Restores an image stored in the given file.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Load(
    string path
)

void Load(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

path

Full path of the file.

serializer

The [ESerializer](#) file-like object that is read from.

Remarks

When loading, an image is resized if need be. On the opposite, an ROI cannot be resized, and the sizes *must* match. The image contents around the ROI remains unchanged.

If a serializer is used, then the Euresys proprietary file format is expected. This format preserves attributes and sub-ROIs.

See Supported Image File Types for details about supported files.

See Image File Access - Save, Load - for details about file format and compatibility.

EBaseROI.OrgX

Gets or sets the x-coordinate of the upper left corner of the ROI, relative to its parent ROI/image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int OrgX
{ get; set; }
```

Remarks

The *placement of an ROI* is given by the coordinates its upper left corner (origin point), relative to its parent ROI/image and by its size (width and height).

EBaseROI.OrgY

Gets or sets the y-coordinate of the upper left corner of the ROI, relative to its parent ROI/image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
int OrgY  
  
    { get; set; }
```

Remarks

The *placement of an ROI* is given by the coordinates its upper left corner (origin point), relative to its parent ROI/image and by its size (width and height).

EBaseROI.Parent

Returns the hierarchical parent of this object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
Euresys.Open_eVision_2_10.EBaseROI Parent  
  
    { get; }
```

EBaseROI.PlanesPerPixel

Gets the number of color components in each pixel of the ROI/image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint PlanesPerPixel  
    { get; }
```

EBaseROI.RowPitch

Gets the pitch of a row (the number of bytes between two vertically adjacent pixels). This is also valid for BW1 images (one bit per pixel).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int RowPitch  
    { get; }
```

EBaseROI.Save

Saves the [EBaseROI](#) object to the given file.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
void Save(
    string path,
    Euresys.Open_eVision_2_10.EImageFileType type
)

void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

path

The full path of the destination file.

type

File format, as defined by [EImageFileType](#). If not specified, the file format is determined from the file extension.

serializer

The [ESerializer](#) file-like object that is written to.

Remarks

By default (if no format is specified), the file format is determined from the file extension.

If a serializer is used, then the Euresys proprietary file format is used. This format preserves attributes and sub-ROIs.

See Supported Image File Types for details about supported files.

See Image File Access - Save, Load - for details about file format and compatibility.

EBaseROI.SaveJpeg

Saves the [EBaseROI](#) object to the given file, in JPEG format.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SaveJpeg(
    string path,
    int quality
)
```

Parameters

path

The full path of the destination file.

quality

JPEG quality, between 0 and 100 (100 is best quality). The default value is 75.

Remarks

See Image File Access - Save, Load - for details about file format and quality.

EBaseROI.SaveJpeg2K

Saves the [EBaseROI](#) object to the given file, in JPEG 2000 format.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SaveJpeg2K(
    string path,
    int quality
)
```

Parameters

path

The full path of the destination file.

quality

JPEG 2000 quality, between 1 and 512. The default value is 16.

Remarks

See Image File Access - Save, Load - for details about file format and quality.

EBaseROI.SavePng

Saves the [EBaseROI](#) object to the given file, in PNG format.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void SavePng(  
    string path,  
    int compression  
)
```

Parameters

path

The full path of the destination file.

compression

PNG compression, between 1 and 9 (1 is the fastest and 9 is the best compression). The default value is 1.

Remarks

See Image File Access - Save, Load - for details about file format and quality.

EBaseROI.Serialize

Serializes the [EBaseROI](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void Serialize(  
    Euresys.Open_eVision_2_10.ESerializer serializer  
)
```

Parameters

serializer

Serializer used for the serialization.

EBaseROI.SetImagePtr

Sets the pointer to an externally allocated image buffer.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetImagePtr(
    int width,
    int height,
    IntPtr imagePointer,
    int bitsPerRow
)
```

Parameters

width

The width of the supplied buffer, in pixels.

height

The height of the supplied buffer, in pixels.

imagePointer

The pointer (aligned on 4 bytes) to the buffer, which must be large enough to hold the image.

bitsPerRow

The total number of bits contained in a row, padding included. Using the value **0** (default) means that this size is computed from the buffer width and the pixel size plus a padding with the smallest possible value that leads to a multiple of 4 bytes (32 bits), which is the minimum padding accepted by [EBaseROI::SetImagePtr](#).

Remarks

This call is only valid on an image. An ROI gets its buffer from its parent while an image normally allocates a pixel buffer automatically. The pointer to this buffer refers to the top left pixel of the image. The next pixels are stored contiguously, row by row, from top to bottom and from left to right.

Padding at the end of a row may be used, but it must lead to rows that are multiple of 4 bytes. This method overrides the internally allocated image buffer of the [EBaseROI](#).

As long as the image accesses this buffer, it must not be deleted.

EBaseROI.SetPlacement

Sets the placement of an ROI, relative to its parent ROI/image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetPlacement(
    int originX,
    int originY,
    int width,
    int height
)
```

Parameters

originX

New x-coordinate of the top-left pixel of this ROI.

originY

New y-coordinate of the top-left pixel of this ROI.

width

New ROI width.

height

New ROI height.

Remarks

This method can only be called on ROIs.

The *placement of an ROI* is given by the coordinates its upper left corner (origin point), relative to its parent ROI/image and by its size (width and height).

EBaseROI.SetSize

Sets the width and height of an ROI/image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetSize(
    int width,
    int height
)

void SetSize(
    Euresys.Open_eVision_2_10.EBaseROI other
)
```

Parameters

width

The new requested ROI/image width.

height

The new requested ROI/image height.

other

The other ROI/image whose dimensions have to be used for the current object.

Remarks

Open eVision will allocate a new image buffer (deallocate the old image buffer) if the supplied width and height are different from the existing ones.

If an external buffer has been specified by means of **SetImagePtr**, it will be kept only if the size does not change.

Creating a new Open eVision image buffer and setting its size creates a 4-byte aligned buffer, by default. The *size of an image* is specified as a number of columns (width) and rows (height).

The maximum image dimensions are 32767 by 32767. Furthermore, it must fit into the available memory, that depends upon the physical memory, the operating system and the memory already allocated by the process in other modules or libraries.

The *placement of an ROI* is given by the x and y coordinates of its upper left pixel relative to its parent image, and also by its width and its height.

EBaseROI.Title

Gets or sets the Title attribute of the ROI/image. This is a convenience property that allows storing some information in the ROI/image. Please note that not all file formats preserve this information.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
string Title
{ get; set; }
```

EBaseROI.TotalHeight

Gets the height, in pixels, of the ROI topmost parent.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int TotalHeight
{ get; }
```

Remarks

The *total* size of an ROI is the size of its *topmost* parent.

EBaseROI.TotalOrgX

Gets the x-coordinate of the upper left pixel of the ROI, relative to its topmost parent upper left pixel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int TotalOrgX
{ get; }
```

Remarks

The *total origin coordinates* of an ROI indicate the position of its upper left pixel relative to the *topmost* parent image. The total origin coordinates (top-left pixel) of a topmost parent are always **(0,0)**.

EBaseROI.TotalOrgY

Gets the y-coordinate of the upper left pixel of the ROI, relative to its topmost parent upper left pixel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int TotalOrgY
{ get; }
```

Remarks

The *total origin coordinates* of an ROI indicate the position of its upper left pixel relative to the *topmost* parent. The total origin coordinates (top-left pixel) of a topmost parent are always **(0,0)**.

EBaseROI.TotalWidth

Gets the width, in pixels, of the ROI topmost parent.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int TotalWidth
{ get; }
```

Remarks

The *total size* of an ROI is the size of its *topmost* parent.

EBaseROI.Type

Gets the ROI/image pixel type, as defined by the [EImageType](#) enumeration.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EImageType Type  
    { get; }
```

EBaseROI.Width

Gets or sets the width of the ROI.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int Width  
    { get; set; }
```

Remarks

The *placement of an ROI* is given by the coordinates its upper left corner (origin point), relative to its parent ROI/image and by its size (width and height).

4.18. EBinaryImageSegmenter Class

Segments a binary image.

Remarks

This segmenter is applicable to [EROIBW1](#) grayscale images.

It produces coded images with two layers: The Black layer (usually, with index 0) contains the unmasked pixels having a binary value equal to zero; and the White layer (usually, with index 1) contains the remaining unmasked pixels, i.e. unmasked pixels having a binary value equal to one.

Base Class: [ETwoLayersImageSegmenter](#)

Namespace: Euresys.Open_eVision_2_10.Segmenters

4.19. EBW16PathVector Class

Vector objects are used to store 1-dimensional data.

Remarks

Using vectors is very similar to using 1-dimensional arrays, except that the size can vary at runtime. Memory allocation is handled internally. * To create a vector, use its constructor. * To fill a vector with values, first empty it, using the [EBW16PathVector](#) member, and then add elements one at time at the tail by calling the [EBW16PathVector::AddElement](#) member. * To access a vector element, either for reading or writing, use the `[]` operator. * To inquire for the current number of elements, use member [EBW16PathVector](#).

Base Class: [EVector](#)

Namespace: Euresys.Open_eVision_2_10

Properties

Closed

-

RawDataPtr

M Pointer to the vector data.

e

Methods

AddElement

Appends (adds at the tail) an element to the vector.

<code>Draw</code>	Draws a plot of the vector element values.
<code>DrawWithCurrentPen</code>	Draws a plot of the vector element values.
<code>EBW16PathVector</code>	Constructs a vector.
<code>GetElement</code>	Returns the vector element at the given index.
<code>operator[]</code>	Gives access to the vector element at the given index.
<code>operator=</code>	Copies all the data from another EBW16PathVector object into the current EBW16PathVector object
<code>SetElement</code>	Modifies the vector element at the given index by the given value.

B

W16PathVector.AddElement

Appends (adds at the tail) an element to the vector.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddElement(
    Euresys.Open_eVision_2_10.EBW16Path element
)
```

Parameters

element

The element to be added.

EBW16PathVector.Closed

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool Closed
{ get; set; }
```

EBW16PathVector.Draw

Draws a plot of the vector element values.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```

void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)

void Draw(
    IntPtr graphicContext,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)

```

Parameters

graphicContext

Handle of the device context on which to draw.

zoomX

Zooming factor along the X axis (**1.0f** means no zoom).

zoomY

Zooming factor along the Y axis (**1.0f** means no zoom).

originX

Abscissa of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

originY

Ordinate of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

color

The color in which to draw the overlay.

Remarks

The vector draws line segment between the element coordinates. The drawing appears on the image itself.

EBW16PathVector.DrawWithCurrentPen

Draws a plot of the vector element values.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawWithCurrentPen(
    IntPtr graphicContext,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

zoomX

Zooming factor along the X axis (**1.0f** means no zoom).

zoomY

Zooming factor along the Y axis (**1.0f** means no zoom).

originX

Abscissa of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

originY

Ordinate of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

Remarks

The vector draws line segment between the element coordinates. The drawing appears on the image itself.

EBW16PathVector.EBW16PathVector

Constructs a vector.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EBW16PathVector (
)
void EBW16PathVector (
    uint maxNumberOfElements
)
void EBW16PathVector (
    Euresys.Open_eVision_2_10.EBW16PathVector other
)
```

Parameters

maxNumberOfElements

Optionally, memory can be pre-allocated to accommodate a given number of elements.

other

EBW16PathVector object to be copied

EBW16PathVector.GetElement

Returns the vector element at the given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EBW16Path GetElement(  
    int index  
)
```

Parameters

index

Index, between **0** and [EBW16PathVector](#) (excluded) of the element to be accessed.

Remarks

If the given index is outside the bounds of the vector, the error code [Parameter1OutOfRange](#) is set.

EBW16PathVector.operator[]

Gives access to the vector element at the given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
ref Euresys.Open_eVision_2_10.EBW16Path operator[] (  
    uint index  
)
```

Parameters

index

Index, between **0** and [EBW16PathVector](#) (excluded) of the element to be accessed.

EBW16PathVector.operator=

Copies all the data from another EBW16PathVector object into the current EBW16PathVector object

Namespace: Euresys.Open_eVision_2_10


```
[C#]
Euresys.Open_eVision_2_10.EBW16PathVector operator=(
    Euresys.Open_eVision_2_10.EBW16PathVector other
)
```

Parameters

other

EBW16PathVector object to be copied

EBW16PathVector.RawDataPtr

Pointer to the vector data.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
IntPtr RawDataPtr
{ get; }
```

EBW16PathVector.SetElement

Modifies the vector element at the given index by the given value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void SetElement(  
    int index,  
    Euresys.Open_eVision_2_10.EBW16Path value  
)
```

Parameters

index

Index, between **0** and [EBW16PathVector](#) (excluded), of the element to be modified.

value

The new value for the element.

Remarks

If the given index is outside the bounds of the vector, the error code [Parameter1OutOfRange](#) is set.

4.20. EBW16PixelAccessor Class

Manages a BW16 pixel accessor context.

Namespace: Euresys.Open_eVision_2_10

Methods

[EBW16PixelAccessor](#)

Constructor.

[GetPixel](#)

Gets the Pixel at the given coordinates.

[SetPixel](#)

Sets the Pixel at the given coordinates.

EBW16PixelAccessor.EBW16PixelAccessor

Constructor.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EBW16PixelAccessor(
    Euresys.Open_eVision_2_10.EROIBW16 roi
)
void EBW16PixelAccessor(
    Euresys.Open_eVision_2_10.EBW16PixelAccessor
)
```

Parameters

roi

Pixel source.

-

EBW16PixelAccessor.GetPixel

Gets the Pixel at the given coordinates.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
ushort GetPixel(
    int x,
    int y
)
```

Parameters

- x*
Pixel X coordinate.
- y*
Pixel Y coordinate.

EBW16PixelAccessor.SetPixel

Sets the Pixel at the given coordinates.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void SetPixel(  
    ushort value,  
    int x,  
    int y  
)
```

Parameters

- value*
Pixel value.
- x*
Pixel X coordinate.
- y*
Pixel Y coordinate.

4.21. EBW16Vector Class

Vector objects are used to store 1-dimensional data.

Remarks

Using vectors is very similar to using 1-dimensional arrays, except that the size can vary at runtime. Memory allocation is handled internally. * To create a vector, use its constructor. * To fill a vector with values, first empty it, using the [EBW16Vector](#) member, and then add elements one at a time at the tail by calling the [EBW16Vector::AddElement](#) member. * To access a vector element, either for reading or writing, use the [] operator. * To inquire for the current number of elements, use member [EBW16Vector](#).

Base Class: [EVector](#)

Namespace: Euresys.Open_eVision_2_10

Properties

[RawDataPtr](#)

M Pointer to the vector data.

e

Methods

[AddElement](#)

Appends (adds at the tail) an element to the vector.

[Draw](#)

Draws a plot of the vector element values.

[DrawWithCurrentPen](#)

Draws a plot of the vector element values.

[EBW16Vector](#)

Constructs a vector.

[GetElement](#)

Returns the vector element at the given index.

[operator\[\]](#)

Gives access to the vector element at the given index.

operator=

Copies all the data from another EBW16Vector object into the current EBW16Vector object

SetElement

Modifies the vector element at the given index by the given value.

WeightedMoment

Returns the first order geometric moment (weighted gravity center).

B

W16Vector.AddElement

Appends (adds at the tail) an element to the vector.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddElement(
    Euresys.Open_eVision_2_10.EBW16 element
)
```

Parameters

element

The element to be added.

EBW16Vector.Draw

Draws a plot of the vector element values.

Namespace: Euresys.Open_eVision_2_10

```

[C#]
void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    float width,
    float height,
    float originX,
    float originY
)

void Draw(
    IntPtr graphicContext,
    float width,
    float height,
    float originX,
    float originY
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    float width,
    float height,
    float originX,
    float originY
)

```

Parameters

graphicContext

Handle of the device context on which to draw.

width

Outermost horizontal size, in pixels.

height

Outermost vertical size, in pixels.

originX

Abscissa of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

originY

Ordinate of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

color

The color in which to draw the overlay.

Remarks

A vector is able to draw itself in a window. The vector plots the element values as a function of the element indices. The drawing appears on a neutral background.

EBW16Vector.DrawWithCurrentPen

Draws a plot of the vector element values.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawWithCurrentPen(
    IntPtr graphicContext,
    float width,
    float height,
    float originX,
    float originY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

width

Outermost horizontal size, in pixels.

height

Outermost vertical size, in pixels.

originX

Abscissa of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

originY

Ordinate of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

Remarks

A vector is able to draw itself in a window. The vector plots the element values as a function of the element indices. The drawing appears on a neutral background.

EBW16Vector.EBW16Vector

Constructs a vector.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EBW16Vector(
)
void EBW16Vector(
    uint maxNumberOfElements
)
void EBW16Vector(
    Euresys.Open_eVision_2_10.EBW16Vector other
)
```

Parameters

maxNumberOfElements

Optionally, memory can be pre-allocated to accommodate a given number of elements.

other

EBW16Vector object to be copied

EBW16Vector.GetElement

Returns the vector element at the given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EBW16 GetElement(  
    int index  
)
```

Parameters

index

Index, between **0** and [EBW16Vector](#) (excluded) of the element to be accessed.

Remarks

If the given index is outside the bounds of the vector, the error code [Parameter1OutOfRange](#) is set.

EBW16Vector.operator[]

Gives access to the vector element at the given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
ref Euresys.Open_eVision_2_10.EBW16 operator[] (  
    uint index  
)
```

Parameters

index

Index, between **0** and [EBW16Vector](#) (excluded) of the element to be accessed.

EBW16Vector.operator=

Copies all the data from another EBW16Vector object into the current EBW16Vector object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EBW16Vector operator=(
    Euresys.Open_eVision_2_10.EBW16Vector other
)
```

Parameters

other

EBW16Vector object to be copied

EBW16Vector.RawDataPtr

Pointer to the vector data.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
IntPtr RawDataPtr
{ get; }
```

EBW16Vector.SetElement

Modifies the vector element at the given index by the given value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void SetElement(  
    int index,  
    Euresys.Open_eVision_2_10.EBW16 value  
)
```

Parameters

index

Index, between **0** and [EBW16Vector](#) (excluded), of the element to be modified.

value

The new value for the element.

Remarks

If the given index is outside the bounds of the vector, the error code [Parameter1OutOfRange](#) is set.

EBW16Vector.WeightedMoment

Returns the first order geometric moment (weighted gravity center).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
float WeightedMoment(  
    uint from,  
    uint to  
)
```

Parameters

from

First element of the vector portion for which the weighted moment will be calculated. By default, this is the first element of the vector.

to

Last element of the vector portion for which the weighted moment will be calculated. By default, this is the last element of the vector.

4.22. EBW32PixelAccessor Class

Manages a BW32 pixel accessor context.

Namespace: Euresys.Open_eVision_2_10

Methods

EBW32PixelAccessor

Constructor.

GetPixel

Gets the Pixel at the given coordinates.

SetPixel

⌈ Sets the Pixel at the given coordinates.

B

W32PixelAccessor.EBW32PixelAccessor

Constructor.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EBW32PixelAccessor(
    Euresys.Open_eVision_2_10.EROIBW32 roi
)
void EBW32PixelAccessor(
    Euresys.Open_eVision_2_10.EBW32PixelAccessor
)
```

Parameters

roi

Pixel source.

-

EBW32PixelAccessor.GetPixel

Gets the Pixel at the given coordinates.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint GetPixel(
    int x,
    int y
)
```

Parameters

x

Pixel X coordinate.

y

Pixel Y coordinate.

EBW32PixelAccessor.SetPixel

Sets the Pixel at the given coordinates.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void SetPixel(  
    uint value,  
    int x,  
    int y  
)
```

Parameters

value

Pixel value.

x

Pixel X coordinate.

y

Pixel Y coordinate.

4.23. EBW32Vector Class

Vector objects are used to store 1-dimensional data.

Remarks

Using vectors is very similar to using 1-dimensional arrays, except that the size can vary at runtime. Memory allocation is handled internally. * To create a vector, use its constructor. * To fill a vector with values, first empty it, using the [EBW32Vector](#) member, and then add elements one at time at the tail by calling the [EBW32Vector::AddElement](#) member. * To access a vector element, either for reading or writing, use the `[]` operator. * To inquire for the current number of elements, use member [EBW32Vector](#).

Base Class: [EVector](#)

Namespace: Euresys.Open_eVision_2_10

Properties

[RawDataPtr](#)

Pointer to the vector data.

Methods

AddElement

Appends (adds at the tail) an element to the vector.

Draw

Draws a plot of the vector element values.

DrawWithCurrentPen

Draws a plot of the vector element values.

EBW32Vector

Constructs a vector.

GetElement

Returns the vector element at the given index.

operator[]

Gives access to the vector element at the given index.

operator=

Copies all the data from another EBW32Vector object into the current EBW32Vector object

SetElement

Modifies the vector element at the given index by the given value.

WeightedMoment

⌈ Returns the first order geometric moment (weighted gravity center).

B

W32Vector.AddElement

Appends (adds at the tail) an element to the vector.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddElement(
    Euresys.Open_eVision_2_10.EBW32 element
)
```

Parameters

element

The element to be added.

EBW32Vector.Draw

Draws a plot of the vector element values.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    float width,
    float height,
    float originX,
    float originY
)

void Draw(
    IntPtr graphicContext,
    float width,
    float height,
    float originX,
    float originY
)
```

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ERGBColor color,  
    float width,  
    float height,  
    float originX,  
    float originY  
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

width

Outermost horizontal size, in pixels.

height

Outermost vertical size, in pixels.

originX

Abscissa of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

originY

Ordinate of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

color

The color in which to draw the overlay.

Remarks

A vector is able to draw itself in a window. The vector plots the element values as a function of the element indices. The drawing appears on a neutral background.

EBW32Vector.DrawWithCurrentPen

Draws a plot of the vector element values.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawWithCurrentPen(
    IntPtr graphicContext,
    float width,
    float height,
    float originX,
    float originY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

width

Outermost horizontal size, in pixels.

height

Outermost vertical size, in pixels.

originX

Abscissa of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

originY

Ordinate of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

Remarks

A vector is able to draw itself in a window. The vector plots the element values as a function of the element indices. The drawing appears on a neutral background.

EBW32Vector.EBW32Vector

Constructs a vector.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EBW32Vector(
)

void EBW32Vector(
    uint maxNumberOfElements
)

void EBW32Vector(
    Euresys.Open_eVision_2_10.EBW32Vector other
)
```

Parameters

maxNumberOfElements

Optionally, memory can be pre-allocated to accommodate a given number of elements.

other

EBW32Vector object to be copied

EBW32Vector.GetElement

Returns the vector element at the given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EBW32 GetElement(
    int index
)
```

Parameters

index

Index, between **0** and [EBW32Vector](#) (excluded) of the element to be accessed.

Remarks

If the given index is outside the bounds of the vector, the error code [Parameter1OutOfRange](#) is set.

EBW32Vector.operator[]

Gives access to the vector element at the given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
ref Euresys.Open_eVision_2_10.EBW32 operator[] (
    uint index
)
```

Parameters

index

Index, between **0** and [EBW32Vector](#) (excluded) of the element to be accessed.

EBW32Vector.operator=

Copies all the data from another EBW32Vector object into the current EBW32Vector object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EBW32Vector operator=(
    Euresys.Open_eVision_2_10.EBW32Vector other
)
```

Parameters

other

EBW32Vector object to be copied

EBW32Vector.RawDataPtr

Pointer to the vector data.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
IntPtr RawDataPtr
{ get; }
```

EBW32Vector.SetElement

Modifies the vector element at the given index by the given value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetElement(
    int index,
    Euresys.Open_eVision_2_10.EBW32 value
)
```

Parameters

index

Index, between **0** and [EBW32Vector](#) (excluded), of the element to be modified.

value

The new value for the element.

Remarks

If the given index is outside the bounds of the vector, the error code [Parameter1OutOfRange](#) is set.

EBW32Vector.WeightedMoment

Returns the first order geometric moment (weighted gravity center).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float WeightedMoment(
    uint from,
    uint to
)
```

Parameters

from

First element of the vector portion for which the weighted moment will be calculated. By default, this is the first element of the vector.

to

Last element of the vector portion for which the weighted moment will be calculated. By default, this is the last element of the vector.

4.24. EBW8PathVector Class

Vector objects are used to store 1-dimensional data.

Remarks

Using vectors is very similar to using 1-dimensional arrays, except that the size can vary at runtime. Memory allocation is handled internally. * To create a vector, use its constructor. * To fill a vector with values, first empty it, using the [EBW8PathVector](#) member, and then add elements one at a time at the tail by calling the [EBW8PathVector::AddElement](#) member. * To access a vector element, either for reading or writing, use the `[]` operator. * To inquire for the current number of elements, use member [EBW8PathVector](#).

Base Class: [EVector](#)

Namespace: Euresys.Open_eVision_2_10

Properties

Closed

Flag indicating whether the shape built with [EasyImage::Contour](#) must be closed or not.

RawDataPtr

M Pointer to the vector data.

e

Methods

AddElement

Appends (adds at the tail) an element to the vector.

Draw

Draws a plot of the vector element values.

DrawWithCurrentPen

Draws a plot of the vector element values.

EBW8PathVector

Constructs a vector.

GetElement

Returns the vector element at the given index.

operator[]

Gives access to the vector element at the given index.

operator=

Copies all the data from another EBW8PathVector object into the current EBW8PathVector object

SetElement

Modifies the vector element at the given index by the given value.

EBW8PathVector.AddElement

Appends (adds at the tail) an element to the vector.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddElement(
    Euresys.Open_eVision_2_10.EBW8Path element
)
```

Parameters

element

The element to be added.

EBW8PathVector.Closed

Flag indicating whether the shape built with [EasyImage::Contour](#) must be closed or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool Closed
{ get; set; }
```

EBW8PathVector.Draw

Draws a plot of the vector element values.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)

void Draw(
    IntPtr graphicContext,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

zoomX

Zooming factor along the X axis (**1.0f** means no zoom).

zoomY

Zooming factor along the Y axis (**1.0f** means no zoom).

originX

Abscissa of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

originY

Ordinate of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

color

The color in which to draw the overlay.

Remarks

The vector draws line segment between the element coordinates. The drawing appears on the image itself.

EBW8PathVector.DrawWithCurrentPen

Draws a plot of the vector element values.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawWithCurrentPen(
    IntPtr graphicContext,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

zoomX

Zooming factor along the X axis (**1.0f** means no zoom).

zoomY

Zooming factor along the Y axis (**1.0f** means no zoom).

originX

Abscissa of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

originY

Ordinate of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

Remarks

The vector draws line segment between the element coordinates. The drawing appears on the image itself.

EBW8PathVector.EBW8PathVector

Constructs a vector.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EBW8PathVector(
)
void EBW8PathVector(
    uint maxNumberOfElements
)
void EBW8PathVector(
    Euresys.Open_eVision_2_10.EBW8PathVector other
)
```

Parameters

maxNumberOfElements

Optionally, memory can be pre-allocated to accommodate a given number of elements.

other

EBW8PathVector object to be copied

EBW8PathVector.GetElement

Returns the vector element at the given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EBW8Path GetElement(  
    int index  
)
```

Parameters

index

Index, between **0** and [EBW8PathVector](#) (excluded) of the element to be accessed.

Remarks

If the given index is outside the bounds of the vector, the error code [Parameter1OutOfRange](#) is set.

EBW8PathVector.operator[]

Gives access to the vector element at the given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
ref Euresys.Open_eVision_2_10.EBW8Path operator[] (  
    uint index  
)
```

Parameters

index

Index, between **0** and [EBW8PathVector](#) (excluded) of the element to be accessed.

EBW8PathVector.operator=

Copies all the data from another EBW8PathVector object into the current EBW8PathVector object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EBW8PathVector operator=(
    Euresys.Open_eVision_2_10.EBW8PathVector other
)
```

Parameters

other

EBW8PathVector object to be copied

EBW8PathVector.RawDataPtr

Pointer to the vector data.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
IntPtr RawDataPtr
{ get; }
```

EBW8PathVector.SetElement

Modifies the vector element at the given index by the given value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void SetElement(  
    int index,  
    Euresys.Open_eVision_2_10.EBW8Path value  
)
```

Parameters

index

Index, between **0** and [EBW8PathVector](#) (excluded), of the element to be modified.

value

The new value for the element.

Remarks

If the given index is outside the bounds of the vector, the error code [Parameter1OutOfRange](#) is set.

4.25. EBW8PixelAccessor Class

Manages a BW8 pixel accessor context.

Namespace: Euresys.Open_eVision_2_10

Methods

[EBW8PixelAccessor](#)

Constructor.

[GetPixel](#)

Gets the Pixel at the given coordinates.

[SetPixel](#)

Sets the Pixel at the given coordinates.

EBW8PixelAccessor.EBW8PixelAccessor

Constructor.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EBW8PixelAccessor(
    Euresys.Open_eVision_2_10.EROIBW8 roi
)
void EBW8PixelAccessor(
    Euresys.Open_eVision_2_10.EBW8PixelAccessor
)
```

Parameters

roi
Pixel source.

-

EBW8PixelAccessor.GetPixel

Gets the Pixel at the given coordinates.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
byte GetPixel(
    int x,
    int y
)
```


Parameters

- x*
Pixel X coordinate.
- y*
Pixel Y coordinate.

EBW8PixelAccessor.SetPixel

Sets the Pixel at the given coordinates.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetPixel(
    byte value,
    int x,
    int y
)
```

Parameters

- value*
Pixel value.
- x*
Pixel X coordinate.
- y*
Pixel Y coordinate.

4.26. EBW8Vector Class

Vector objects are used to store 1-dimensional data.

Remarks

Using vectors is very similar to using 1-dimensional arrays, except that the size can vary at runtime. Memory allocation is handled internally. * To create a vector, use its constructor. * To fill a vector with values, first empty it, using the [EBW8Vector](#) member, and then add elements one at a time at the tail by calling the [EBW8Vector::AddElement](#) member. * To access a vector element, either for reading or writing, use the `[]` operator. * To inquire for the current number of elements, use member [EBW8Vector](#).

Base Class: [EVector](#)

Namespace: Euresys.Open_eVision_2_10

Properties

[RawDataPtr](#)

M Pointer to the vector data.

e

Methods

[AddElement](#)

Appends (adds at the tail) an element to the vector.

[Draw](#)

Draws a plot of the vector element values.

[DrawWithCurrentPen](#)

Draws a plot of the vector element values.

[EBW8Vector](#)

Constructs a vector.

[GetElement](#)

Returns the vector element at the given index.

[operator\[\]](#)

Gives access to the vector element at the given index.

operator=

Copies all the data from another EBW8Vector object into the current EBW8Vector object

SetElement

Modifies the vector element at the given index by the given value.

WeightedMoment

Returns the first order geometric moment (weighted gravity center).

B

W8Vector.AddElement

Appends (adds at the tail) an element to the vector.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddElement(
    Euresys.Open_eVision_2_10.EBW8 element
)
```

Parameters

element

The element to be added.

EBW8Vector.Draw

Draws a plot of the vector element values.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void Draw(  
    IntPtr graphicContext,  
    float width,  
    float height,  
    float originX,  
    float originY  
)  
  
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    float width,  
    float height,  
    float originX,  
    float originY  
)  
  
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ERGBColor color,  
    float width,  
    float height,  
    float originX,  
    float originY  
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

width

Outermost horizontal size, in pixels.

height

Outermost vertical size, in pixels.

originX

Abscissa of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

originY

Ordinate of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

color

The color in which to draw the overlay.

Remarks

A vector is able to draw itself in a window. The vector plots the element values as a function of the element indices. The drawing appears on a neutral background.

EBW8Vector.DrawWithCurrentPen

Draws a plot of the vector element values.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawWithCurrentPen(
    IntPtr graphicContext,
    float width,
    float height,
    float originX,
    float originY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

width

Outermost horizontal size, in pixels.

height

Outermost vertical size, in pixels.

originX

Abscissa of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

originY

Ordinate of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

Remarks

A vector is able to draw itself in a window. The vector plots the element values as a function of the element indices. The drawing appears on a neutral background.

EBW8Vector.EBW8Vector

Constructs a vector.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EBW8Vector(
)
void EBW8Vector(
    uint maxNumberOfElements
)
void EBW8Vector(
    Euresys.Open_eVision_2_10.EBW8Vector other
)
```

Parameters

maxNumberOfElements

Optionally, memory can be pre-allocated to accommodate a given number of elements.

other

EBW8Vector object to be copied

EBW8Vector.GetElement

Returns the vector element at the given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EBW8 GetElement(  
    int index  
)
```

Parameters

index

Index, between **0** and [EBW8Vector](#) (excluded) of the element to be accessed.

Remarks

If the given index is outside the bounds of the vector, the error code [Parameter1OutOfRange](#) is set.

EBW8Vector.operator[]

Gives access to the vector element at the given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
ref Euresys.Open_eVision_2_10.EBW8 operator[](  
    uint index  
)
```

Parameters

index

Index, between **0** and [EBW8Vector](#) (excluded) of the element to be accessed.

EBW8Vector.operator=

Copies all the data from another EBW8Vector object into the current EBW8Vector object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EBW8Vector operator=(
    Euresys.Open_eVision_2_10.EBW8Vector other
)
```

Parameters

other

EBW8Vector object to be copied

EBW8Vector.RawDataPtr

Pointer to the vector data.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
IntPtr RawDataPtr
{ get; }
```

EBW8Vector.SetElement

Modifies the vector element at the given index by the given value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```



```
void SetElement(  
    int index,  
    Euresys.Open_eVision_2_10.EBW8 value  
)
```

Parameters

index

Index, between **0** and [EBW8Vector](#) (excluded), of the element to be modified.

value

The new value for the element.

Remarks

If the given index is outside the bounds of the vector, the error code [Parameter1OutOfRange](#) is set.

EBW8Vector.WeightedMoment

Returns the first order geometric moment (weighted gravity center).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
float WeightedMoment(  
    uint from,  
    uint to  
)
```

Parameters

from

First element of the vector portion for which the weighted moment will be calculated. By default, this is the first element of the vector.

to

Last element of the vector portion for which the weighted moment will be calculated. By default, this is the last element of the vector.

4.27. EBWHistogramVector Class

Vector objects are used to store 1-dimensional data.

Remarks

Using vectors is very similar to using 1-dimensional arrays, except that the size can vary at runtime. Memory allocation is handled internally. * To create a vector, use its constructor. * To fill a vector with values, first empty it, using the [EBWHistogramVector](#) member, and then add elements one at time at the tail by calling the [EBWHistogramVector::AddElement](#) member. * To access a vector element, either for reading or writing, use the [] operator. * To inquire for the current number of elements, use member [EBWHistogramVector](#).

Base Class: [EVector](#)

Namespace: Euresys.Open_eVision_2_10

Properties

[RawDataPtr](#)

M Pointer to the vector data.

e

Methods

[AddElement](#)

Appends (adds at the tail) an element to the vector.

[Draw](#)

Draws a plot of the vector element values.

[DrawWithCurrentPen](#)

Draws a plot of the vector element values.

[EBWHistogramVector](#)

Constructs a vector.

GetElement

Returns the vector element at the given index.

operator[]

Gives access to the vector element at the given index.

operator=

Copies all the data from another EBWHistogramVector object into the current EBWHistogramVector object

SetElement

Modifies the vector element at the given index by the given value.

B

WHistogramVector.AddElement

Appends (adds at the tail) an element to the vector.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void AddElement(  
    uint element  
)
```

Parameters

element

The element to be added.

EBWHistogramVector.Draw

Draws a plot of the vector element values.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    float width,
    float height,
    float originX,
    float originY
)

void Draw(
    IntPtr graphicContext,
    float width,
    float height,
    float originX,
    float originY
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    float width,
    float height,
    float originX,
    float originY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

width

Outermost horizontal size, in pixels.

height

Outermost vertical size, in pixels.

originX

Abcissa of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

originY

Ordinate of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

color

The color in which to draw the overlay.

Remarks

A vector is able to draw itself in a window. The vector plots the element values as a function of the element indices. The drawing appears on a neutral background.

EBWHistogramVector.DrawWithCurrentPen

Draws a plot of the vector element values.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawWithCurrentPen(
    IntPtr graphicContext,
    float width,
    float height,
    float originX,
    float originY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

width

Outermost horizontal size, in pixels.

height

Outermost vertical size, in pixels.

originX

Abscissa of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

originY

Ordinate of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

Remarks

A vector is able to draw itself in a window. The vector plots the element values as a function of the element indices. The drawing appears on a neutral background.

EBWHistogramVector.EBWHistogramVector

Constructs a vector.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EBWHistogramVector(
)
void EBWHistogramVector(
    Euresys.Open_eVision_2_10.EBWHistogramVector other
)
void EBWHistogramVector(
    uint maxNumberOfElements
)
```

Parameters

other

EBWHistogramVector object to be copied

maxNumberOfElements

Optionally, memory can be pre-allocated to accommodate a given number of elements.

EBWHistogramVector.GetElement

Returns the vector element at the given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
uint GetElement(  
    int index  
)
```

Parameters

index

Index, between **0** and [EBWHistogramVector](#) (excluded) of the element to be accessed.

Remarks

If the given index is outside the bounds of the vector, the error code [Parameter1OutOfRange](#) is set.

EBWHistogramVector.operator[]

Gives access to the vector element at the given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
ref uint operator[](  
    uint index  
)
```

Parameters

index

Index, between **0** and [EBWHistogramVector](#) (excluded) of the element to be accessed.

EBWHistogramVector.operator=

Copies all the data from another EBWHistogramVector object into the current EBWHistogramVector object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EBWHistogramVector operator=(
    Euresys.Open_eVision_2_10.EBWHistogramVector other
)
```

Parameters

other

EBWHistogramVector object to be copied

EBWHistogramVector.RawDataPtr

Pointer to the vector data.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
IntPtr RawDataPtr
{ get; }
```

EBWHistogramVector.SetElement

Modifies the vector element at the given index by the given value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void SetElement(  
    int index,  
    uint value  
)
```

Parameters

index

Index, between **0** and [EBWHistogramVector](#) (excluded), of the element to be modified.

value

The new value for the element.

Remarks

If the given index is outside the bounds of the vector, the error code [Parameter1OutOfRange](#) is set.

4.28. EC15PixelAccessor Class

Manages a C15 pixel accessor context.

Namespace: Euresys.Open_eVision_2_10

Methods

[EC15PixelAccessor](#)

Constructor.

[GetPixel](#)

Gets the Pixel at the given coordinates.

[SetPixel](#)

Sets the Pixel at the given coordinates.

EC15PixelAccessor.EC15PixelAccessor

Constructor.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EC15PixelAccessor(
    Euresys.Open_eVision_2_10.EROIC15 roi
)
void EC15PixelAccessor(
    Euresys.Open_eVision_2_10.EC15PixelAccessor
)
```

Parameters

roi
Pixel source.

-

EC15PixelAccessor.GetPixel

Gets the Pixel at the given coordinates.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EC15 GetPixel(
    int x,
    int y
)
```

Parameters

- x*
Pixel X coordinate.
- y*
Pixel Y coordinate.

EC15PixelAccessor.SetPixel

Sets the Pixel at the given coordinates.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetPixel(
    Euresys.Open_eVision_2_10.EC15 value,
    int x,
    int y
)
```

Parameters

- value*
Pixel value.
- x*
Pixel X coordinate.
- y*
Pixel Y coordinate.

4.29. EC16PixelAccessor Class

Manages a C16 pixel accessor context.

Namespace: Euresys.Open_eVision_2_10

Methods

EC16PixelAccessor

Constructor.

GetPixel

Gets the Pixel at the given coordinates.

SetPixel

⌊ Sets the Pixel at the given coordinates.

C

16PixelAccessor.EC16PixelAccessor

Constructor.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EC16PixelAccessor(
    Euresys.Open_eVision_2_10.EROIC16 roi
)
void EC16PixelAccessor(
    Euresys.Open_eVision_2_10.EC16PixelAccessor
)
```

Parameters

roi

Pixel source.

-

EC16PixelAccessor.GetPixel

Gets the Pixel at the given coordinates.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EC16 GetPixel(
    int x,
    int y
)
```

Parameters

x
Pixel X coordinate.

y
Pixel Y coordinate.

EC16PixelAccessor.SetPixel

Sets the Pixel at the given coordinates.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetPixel(
    Euresys.Open_eVision_2_10.EC16 value,
    int x,
    int y
)
```

Parameters

value

Pixel value.

x

Pixel X coordinate.

y

Pixel Y coordinate.

4.30. EC24APixelAccessor Class

Manages a C24A pixel accessor context.

Namespace: Euresys.Open_eVision_2_10

Methods

EC24APixelAccessor

Constructor.

GetPixel

Gets the Pixel at the given coordinates.

SetPixel

⌈ Sets the Pixel at the given coordinates.

C

24APixelAccessor.EC24APixelAccessor

Constructor.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EC24APixelAccessor(
    Euresys.Open_eVision_2_10.EROIC24A roi
)

void EC24APixelAccessor(
    Euresys.Open_eVision_2_10.EC24APixelAccessor
)
```

Parameters

roi

Pixel source.

-

EC24APixelAccessor.GetPixel

Gets the Pixel at the given coordinates.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EC24A GetPixel(
    int x,
    int y
)
```

Parameters

x

Pixel X coordinate.

y

Pixel Y coordinate.

EC24APixelAccessor.SetPixel

Sets the Pixel at the given coordinates.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetPixel(
    Euresys.Open_eVision_2_10.EC24A value,
    int x,
    int y
)
```

Parameters

value

Pixel value.

x

Pixel X coordinate.

y

Pixel Y coordinate.

4.31. EC24PathVector Class

Vector objects are used to store 1-dimensional data.

Remarks

Using vectors is very similar to using 1-dimensional arrays, except that the size can vary at runtime. Memory allocation is handled internally. * To create a vector, use its constructor. * To fill a vector with values, first empty it, using the [EC24PathVector](#) member, and then add elements one at time at the tail by calling the [EC24PathVector::AddElement](#) member. * To access a vector element, either for reading or writing, use the `[]` operator. * To inquire for the current number of elements, use member [EC24PathVector](#).

Base Class: [EVector](#)

Namespace: Euresys.Open_eVision_2_10

Properties

Closed

-

RawDataPtr

M Pointer to the vector data.

e

Methods

AddElement

Appends (adds at the tail) an element to the vector.

Draw

Draws a plot of the vector element values.

DrawWithCurrentPen

Draws a plot of the vector element values.

EC24PathVector

Constructs a vector.

GetElement

Returns the vector element at the given index.

operator[]

Gives access to the vector element at the given index.

operator=

Copies all the data from another EC24PathVector object into the current EC24PathVector object

SetElement

Modifies the vector element at the given index by the given value.

EC24PathVector.AddElement

Appends (adds at the tail) an element to the vector.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddElement(
    Euresys.Open_eVision_2_10.EC24Path element
)
```

Parameters

element

The element to be added.

EC24PathVector.Closed

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool Closed
{ get; set; }
```

EC24PathVector.Draw

Draws a plot of the vector element values.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)

void Draw(
    IntPtr graphicContext,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

zoomX

Zooming factor along the X axis (**1.0f** means no zoom).

zoomY

Zooming factor along the Y axis (**1.0f** means no zoom).

originX

Abscissa of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

originY

Ordinate of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

color

The color in which to draw the overlay.

Remarks

The vector draws line segment between the element coordinates. The drawing appears on the image itself.

EC24PathVector.DrawWithCurrentPen

Draws a plot of the vector element values.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawWithCurrentPen(
    IntPtr graphicContext,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

zoomX

Zooming factor along the X axis (**1.0f** means no zoom).

zoomY

Zooming factor along the Y axis (**1.0f** means no zoom).

originX

Abscissa of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

originY

Ordinate of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

Remarks

The vector draws line segment between the element coordinates. The drawing appears on the image itself.

EC24PathVector.EC24PathVector

Constructs a vector.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EC24PathVector(
)
void EC24PathVector(
    Euresys.Open_eVision_2_10.EC24PathVector other
)
void EC24PathVector(
    uint maxNumberOfElements
)
```

Parameters

other

EC24PathVector object to be copied

maxNumberOfElements

Optionally, memory can be pre-allocated to accommodate a given number of elements.

EC24PathVector.GetElement

Returns the vector element at the given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EC24Path GetElement(  
    int index  
)
```

Parameters

index

Index, between **0** and [EC24PathVector](#) (excluded) of the element to be accessed.

Remarks

If the given index is outside the bounds of the vector, the error code [Parameter1OutOfRange](#) is set.

EC24PathVector.operator[]

Gives access to the vector element at the given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
ref Euresys.Open_eVision_2_10.EC24Path operator[] (  
    uint index  
)
```

Parameters

index

Index, between **0** and [EC24PathVector](#) (excluded) of the element to be accessed.

EC24PathVector.operator=

Copies all the data from another EC24PathVector object into the current EC24PathVector object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EC24PathVector operator=(
    Euresys.Open_eVision_2_10.EC24PathVector other
)
```

Parameters

other

EC24PathVector object to be copied

EC24PathVector.RawDataPtr

Pointer to the vector data.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
IntPtr RawDataPtr
{ get; }
```

EC24PathVector.SetElement

Modifies the vector element at the given index by the given value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void SetElement(  
    int index,  
    Euresys.Open_eVision_2_10.EC24Path value  
)
```

Parameters

index

Index, between **0** and [EC24PathVector](#) (excluded), of the element to be modified.

value

The new value for the element.

Remarks

If the given index is outside the bounds of the vector, the error code [Parameter1OutOfRange](#) is set.

4.32. EC24PixelAccessor Class

Manages a C24 pixel accessor context.

Namespace: Euresys.Open_eVision_2_10

Methods

[EC24PixelAccessor](#)

Constructor.

[GetPixel](#)

Gets the Pixel at the given coordinates.

[SetPixel](#)

Sets the Pixel at the given coordinates.

EC24PixelAccessor.EC24PixelAccessor

Constructor.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EC24PixelAccessor(
    Euresys.Open_eVision_2_10.EROIC24 roi
)
void EC24PixelAccessor(
    Euresys.Open_eVision_2_10.EC24PixelAccessor
)
```

Parameters

roi
Pixel source.

-

EC24PixelAccessor.GetPixel

Gets the Pixel at the given coordinates.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EC24 GetPixel(
    int x,
    int y
)
```

Parameters

- x*
Pixel X coordinate.
- y*
Pixel Y coordinate.

EC24PixelAccessor.SetPixel

Sets the Pixel at the given coordinates.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetPixel(
    Euresys.Open_eVision_2_10.EC24 value,
    int x,
    int y
)
```

Parameters

- value*
Pixel value.
- x*
Pixel X coordinate.
- y*
Pixel Y coordinate.

4.33. EC24Vector Class

Vector objects are used to store 1-dimensional data.

Remarks

Using vectors is very similar to using 1-dimensional arrays, except that the size can vary at runtime. Memory allocation is handled internally. * To create a vector, use its constructor. * To fill a vector with values, first empty it, using the [EC24Vector](#) member, and then add elements one at a time at the tail by calling the [EC24Vector::AddElement](#) member. * To access a vector element, either for reading or writing, use the [] operator. * To inquire for the current number of elements, use member [EC24Vector](#).

Base Class: [EVector](#)

Namespace: Euresys.Open_eVision_2_10

Properties

[RawDataPtr](#)

M Pointer to the vector data.

e

Methods

[AddElement](#)

Appends (adds at the tail) an element to the vector.

[Draw](#)

Draws a plot of the vector element values.

[EC24Vector](#)

Constructs a vector.

[GetElement](#)

Returns the vector element at the given index.

[operator\[\]](#)

Gives access to the vector element at the given index.

[operator=](#)

Copies all the data from another [EC24Vector](#) object into the current [EC24Vector](#) object

SetElement

Modifies the vector element at the given index by the given value.

C

24Vector.AddElement

Appends (adds at the tail) an element to the vector.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void AddElement(  
    Euresys.Open_eVision_2_10.EC24 element  
)
```

Parameters

element

The element to be added.

EC24Vector.Draw

Draws a plot of the vector element values.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void Draw(  
    IntPtr graphicContext,  
    float width,  
    float height  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    float width,  
    float height,  
    float originX,  
    float originY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    float width,  
    float height,  
    float originX,  
    float originY,  
    Euresys.Open_eVision_2_10.ERGBColor color0,  
    Euresys.Open_eVision_2_10.ERGBColor color1,  
    Euresys.Open_eVision_2_10.ERGBColor color2  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    float width,  
    float height,  
    Euresys.Open_eVision_2_10.ERGBColor color0,  
    Euresys.Open_eVision_2_10.ERGBColor color1,  
    Euresys.Open_eVision_2_10.ERGBColor color2  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    float width,  
    float height  
)
```

```

void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    float width,
    float height,
    float originX,
    float originY
)

void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    float width,
    float height,
    float originX,
    float originY,
    Euresys.Open_eVision_2_10.ERGBColor color0,
    Euresys.Open_eVision_2_10.ERGBColor color1,
    Euresys.Open_eVision_2_10.ERGBColor color2
)

void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    float width,
    float height,
    Euresys.Open_eVision_2_10.ERGBColor color0,
    Euresys.Open_eVision_2_10.ERGBColor color1,
    Euresys.Open_eVision_2_10.ERGBColor color2
)

```

Parameters

graphicContext

Handle of the device context on which to draw.

width

Outermost horizontal size, in pixels.

height

Outermost vertical size, in pixels.

originX

Abscissa of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

originY

Ordinate of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

color0

The color to be used when drawing the curve of the first color component of the vector, as an RGB color. By default, the current pen is used to draw the curve.

color1

The color to be used when drawing the curve of the second color component of the vector, as an RGB color. By default, the current pen is used to draw the curve.

color2

The color to be used when drawing the curve of the third color component of the vector, as an RGB color. By default, the current pen is used to draw the curve.

Remarks

A vector is able to draw itself in a window. The vector plots the element values as a function of the element indices. The drawing appears on a neutral background. In the special case of the EC24Vector, three curves are drawn instead of one, each corresponding to a color component. Three pen objects must be provided to draw the curves with appropriate attributes.

EC24Vector.EC24Vector

Constructs a vector.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EC24Vector(
)
void EC24Vector(
    uint maxNumberOfElements
)
void EC24Vector(
    Euresys.Open_eVision_2_10.EC24Vector other
)
```

Parameters

maxNumberOfElements

Optionally, memory can be pre-allocated to accommodate a given number of elements.

other

EC24Vector object to be copied

EC24Vector.GetElement

Returns the vector element at the given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EC24 GetElement(
    int index
)
```

Parameters

index

Index, between **0** and [EC24Vector](#) (excluded) of the element to be accessed.

Remarks

If the given index is outside the bounds of the vector, the error code [Parameter1OutOfRange](#) is set.

EC24Vector.operator[]

Gives access to the vector element at the given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
ref Euresys.Open_eVision_2_10.EC24 operator[] (
    uint index
)
```

Parameters

index

Index, between **0** and [EC24Vector](#) (excluded) of the element to be accessed.

EC24Vector.operator=

Copies all the data from another EC24Vector object into the current EC24Vector object

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EC24Vector operator=(  
    Euresys.Open_eVision_2_10.EC24Vector other  
)
```

Parameters

other

EC24Vector object to be copied

EC24Vector.RawDataPtr

Pointer to the vector data.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
IntPtr RawDataPtr  
{ get; }
```

EC24Vector.SetElement

Modifies the vector element at the given index by the given value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetElement(
    int index,
    Euresys.Open_eVision_2_10.EC24 value
)
```

Parameters

index

Index, between **0** and [EC24Vector](#) (excluded), of the element to be modified.

value

The new value for the element.

Remarks

If the given index is outside the bounds of the vector, the error code [Parameter1OutOfRange](#) is set.

4.34. ECalibrationGenerator Class

Represents a 3D calibration model generator, a class made to compute calibration models.

Derived Class(es): [EObjectBasedCalibrationGenerator](#)

Namespace: Euresys.Open_eVision_2_10.Easy3D

4.35. ECalibrationModel Class

Represents a 3D calibration model.

Derived Class(es): [EExplicitGeometricCalibrationModel](#) [EObjectBasedCalibrationModel](#) [EScaleCalibrationModel](#)

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

Type **M** Returns the type of calibration model, see [ECalibrationType](#).

Methods

Apply **e**
Applies this model to convert an uncalibrated point to a world position.
This method returns a world position.

Create
Factory method from a serializer stream: allocates and reads the calibration model from the given serializer.
Returns the corresponding calibration model, must be released by caller.

Save
ECalibrationModel.Apply
Saves the calibration model. The given **ESerializer** must have been created by this model.
Applies this model to convert an uncalibrated point to a world position.
This method returns a world position.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DPoint Apply(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint uvwPoint
)
```

Parameters

uvwPoint

The position of a depth map pixel.

ECalibrationModel.Create

Factory method from a serializer stream: allocates and reads the calibration model from the given serializer. Returns the corresponding calibration model, must be released by caller.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.Easy3D.ECalibrationModel Create(  
    Euresys.Open_eVision_2_10.ESerializer file  
)
```

Parameters

file

A serializer created for reading.

ECalibrationModel.Save

Saves the calibration model. The given ESerializer must have been created for writing.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void Save(  
    Euresys.Open_eVision_2_10.ESerializer serializer  
)
```

Parameters

serializer

ECalibrationModel.Type

Returns the type of calibration model, see [ECalibrationType](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
virtual Euresys.Open_eVision_2_10.Easy3D.ECalibrationType Type  
{ get; }
```

4.36. ECannyEdgeDetector Class

Manages a complete context for the Canny edge detector.

Remarks

The Canny edge detector operates on a grayscale BW8 image and delivers a black-and-white BW8 image where pixels have only 2 possible values: 0 and 255. Pixels corresponding to edges in the source image are set to value 255 in the output image; The other pixels are set to value 0.

Namespace: Euresys.Open_eVision_2_10

Properties

HighThreshold

Sets the high hysteresis threshold for a pixel to be considered as an edge.

LowThreshold

Sets the low hysteresis threshold for a pixel to be considered as an edge.

SmoothingScale

The scale of the features of interest.

ThresholdingMode

M Sets the mode of the hysteresis thresholding.

e

Methods

Apply

Apply the Canny edge detector on an image/ROI.

ECannyEdgeDetector

Constructs a ECannyEdgeDetector object initialized to its default values.

ResetSmoothingScale

E Prevents the smoothing of the source image by a Gaussian filter.

C

CannyEdgeDetector.Apply

Apply the Canny edge detector on an image/ROI.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void Apply(  
    Euresys.Open_eVision_2_10.EROIBW8 source,  
    Euresys.Open_eVision_2_10.EROIBW8 result  
)
```

Parameters

source

The source image/ROI.

result

The output image/ROI.

Remarks

The output ROI must have the same size than the input ROI.

ECannyEdgeDetector.ECannyEdgeDetector

Constructs a ECannyEdgeDetector object initialized to its default values.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ECannyEdgeDetector (
    Euresys.Open_eVision_2_10.ECannyEdgeDetector other
)
void ECannyEdgeDetector (
)
```

Parameters

other

-

ECannyEdgeDetector.HighThreshold

Sets the high hysteresis threshold for a pixel to be considered as an edge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float HighThreshold
{ get; set; }
```

ECannyEdgeDetector.LowThreshold

Sets the low hysteresis threshold for a pixel to be considered as an edge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float LowThreshold
{ get; set; }
```

ECannyEdgeDetector.ResetSmoothingScale

Prevents the smoothing of the source image by a Gaussian filter.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ResetSmoothingScale(
)
```

Remarks

Calling this method is equivalent to set [ECannyEdgeDetector::SmoothingScale](#) to zero. It disables the use of the Gaussian filter.

ECannyEdgeDetector.SmoothingScale

The scale of the features of interest.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float SmoothingScale  
  
{ get; set; }
```

Remarks

This scale corresponds to the standard deviation of the Gaussian filter that is used to smooth the source image before the computation of the gradient, hereby selecting the scale of the features of interest.

If this scale is set to zero, no smoothing is achieved: The gradient is computed directly on the raw source image, speeding up the detector, but making the process much less reliable.

ECannyEdgeDetector.ThresholdingMode

Sets the mode of the hysteresis thresholding.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.ECannyThresholdingMode ThresholdingMode  
  
{ get; set; }
```

Remarks

If the threshold mode is set to [Absolute](#), the threshold values are interpreted as absolute thresholds. In this case, the thresholds must be strictly positive real values.

If the threshold mode is set to [Relative](#), the thresholds are expressed as a fraction ranging from 0 to 1 of the maximum value of the gradient of the source image.

In either case, the low threshold must be less than the high threshold.

4.37. EChecker Class

Manages a complete context for the inspection tool based on image comparison in EasyOCV.

Namespace: Euresys.Open_eVision_2_10

Properties

[Average](#)

Global intensity of the mother image.

[DegreesOfFreedom](#)

Boolean combination of [EDegreesOfFreedom](#) members, that indicates which degrees of freedom are to be considered.

[Deviation](#)

Global contrast of the mother image.

[High](#)

High threshold image for the adaptive segmentation.

[HitHandle](#)

Handle currently hit.

[HitRoi](#)

ROI currently hit.

Low	Low threshold image for the adaptive segmentation.
Normalize	Current normalization mode.
NumAverageSamples	Number of samples that were accumulated in the "average" phase of the training.
NumDeviationSamples	Number of samples that were accumulated in the "deviation" phase of the training.
PanX	Current horizontal panning factor for use in display operations.
PanY	Current vertical panning factor for use in display operations.
Registered	Represents the source image, after it has been aligned with the reference image.
RelativeTolerance	Current tolerance factor to be used for threshold image setup.
ToleranceX	Current horizontal search tolerance, in pixels.
ToleranceY	Current vertical search tolerance, in pixels.

ZoomX

Current horizontal zooming factor for use in display operations.

ZoomY

M Current vertical zooming factor for use in display operations.

e

thods

AddPathName

Adds a single file pathname.

Attach

Associates a source image to a checker context.

BatchLearn

Performs the learning sequence using the specified list of image files.

Drag

Moves the relevant ROI by means of its handle.

Draw

Draws one of the geometric items that define the [EChecker](#) tool.

DrawWithCurrentPen

Draws one of the geometric items that define the [EChecker](#) tool.

EChecker

Constructs an uninitialized checker context.

EmptyPathNames

Clears the list of file pathnames.

HitTest

Returns **TRUE** if the cursor is over one of the dragging handles.

Learn	Accumulates a reference image, following a sequence of operations.
Load	-
Register	Realigns and normalizes the source image.
Save	-
SetPan	Sets the panning factor for use in display operations.
SetTolerance	Sets the search margin, i.e. the width and height of the search area surrounding the alignment pattern(s).
SetZoom	Sets the zooming factor for use in display operations.

C

hecker.AddPathName

Adds a single file pathname.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void AddPathName (  
    string pathName  
)
```

Parameters

pathName

NULL terminated text string containing the file pathname.

EChecker.Attach

Associates a source image to a checker context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Attach(  
    Euresys.Open_eVision_2_10.EROIBW8 source  
)
```

Parameters

source

Pointer to the source image.

Remarks

The source image is used in all consecutive learning/inspection operations.

EChecker.Average

Global intensity of the mother image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float Average
{ get; }
```

Remarks

Valid in mode [Moments](#) only.

EChecker.BatchLearn

Performs the learning sequence using the specified list of image files.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void BatchLearn(
    Euresys.Open_eVision_2_10.ELearningMode mode
)
```

Parameters

mode

[RmsDeviation](#) or [AbsDeviation](#), depending on the preferred method of computing the deviations.

EChecker.DegreesOfFreedom

Boolean combination of [EDegreesOfFreedom](#) members, that indicates which degrees of freedom are to be considered.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint DegreesOfFreedom
    { get; set; }
```

EChecker.Deviation

Global contrast of the mother image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float Deviation
    { get; }
```

Remarks

Valid in mode [Moments](#) only.

EChecker.Drag

Moves the relevant ROI by means of its handle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Drag(
    int x,
    int y
)
```


Parameters

x

New horizontal cursor position.

y

New vertical cursor position.

EChecker.Draw

Draws one of the geometric items that define the [EChecker](#) tool.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,  
    bool handles,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ERGBColor color,  
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,  
    bool handles,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,  
    bool handles,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

Parameters

graphicContext

Device context of the drawing window.

drawingMode

ROI to be drawn, as defined by [EDrawingMode](#).

handles

TRUE if the dragging handles must be displayed.

zoomX

Magnification factor for zooming in or out in the horizontal direction. By default, the image is displayed in true scale.

zoomY

Magnification factor for zooming in or out in the vertical direction. Setting a **0** value (which is the default) will result in isotropic scaling (i.e. equal horizontal and vertical factors).

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

color

The color in which to draw the overlay.

EChecker.DrawWithCurrentPen

Draws one of the geometric items that define the [EChecker](#) tool.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawWithCurrentPen(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool handles,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

Parameters

graphicContext

Device context of the drawing window.

drawingMode

ROI to be drawn, as defined by [EDrawingMode](#).

handles

TRUE if the dragging handles must be displayed.

zoomX

Magnification factor for zooming in or out in the horizontal direction. By default, the image is displayed in true scale.

zoomY

Magnification factor for zooming in or out in the vertical direction. Setting a **0** value (which is the default) will result in isotropic scaling (i.e. equal horizontal and vertical factors).

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

EChecker.EChecker

Constructs an uninitialized checker context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EChecker (
    Euresys.Open_eVision_2_10.EChecker other
)
void EChecker (
)
```

Parameters

other

-

EChecker.EmptyPathNames

Clears the list of file pathnames.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EmptyPathNames (
)
```

EChecker.High

High threshold image for the adaptive segmentation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EImageBW8 High
```

```
{ get; }
```

EChecker.HitHandle

Handle currently hit.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EDragHandle HitHandle
```

```
{ get; }
```

EChecker.HitRoi

ROI currently hit.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.ERoiHit HitRoi
```

```
{ get; }
```

EChecker.HitTest

Returns **TRUE** if the cursor is over one of the dragging handles.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool HitTest(
    int x,
    int y
)
```

Parameters

x
Current horizontal cursor position.

y
Current vertical cursor position.

Remarks

In this case, [EChecker::HitRoi](#) returns the name of the ROI that has been hit, and [EChecker::HitHandle](#) returns the name of the corresponding handle.

EChecker.Learn

Accumulates a reference image, following a sequence of operations.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void Learn(  
    Euresys.Open_eVision_2_10.ELearningMode mode  
)
```

Parameters

mode

Current mode of operation in the learning sequence, as defined by [ELearningMode](#).

Remarks

First the model is reset; then the matching patterns are shown; next a series of images is presented to estimate the average gray levels; then a second series of images is presented to estimate the gray-level variations; finally, the threshold images are generated. A typical sequence with three reference images goes as follows: For standard deviation estimation [EChecker.Learn\(Reset\)](#); initializes. [EChecker.Register\(\)](#); realigns and normalizes 1st source image. [EChecker.Learn\(RmsDeviation\)](#); processes 1st image for deviation info. [EChecker.Register\(\)](#); realigns and normalizes 2nd source image. [EChecker.Learn\(RmsDeviation\)](#); processes 2nd image for deviation info. [EChecker.Register\(\)](#); realigns and normalizes 3rd source image. [EChecker.Learn\(RmsDeviation\)](#); processes 3rd image for deviation info. For robust deviation estimation [EChecker.Learn\(Reset\)](#); initializes. [EChecker.Register\(\)](#); realigns and normalizes 1st source image. [EChecker.Learn\(Average\)](#); processes 1st image for average info. [EChecker.Register\(\)](#); realigns and normalizes 2nd source image. [EChecker.Learn\(Average\)](#); processes 2nd image for average info. [EChecker.Register\(\)](#); realigns and normalizes 3rd source image. [EChecker.Learn\(Average\)](#); processes 3rd image for average info. [EChecker.Register\(\)](#); realigns and normalizes 1st source image. [EChecker.Learn\(RmsDeviation\)](#); processes 1st image for deviation info. [EChecker.Register\(\)](#); realigns and normalizes 2nd source image. [EChecker.Learn\(RmsDeviation\)](#); processes 2nd image for deviation info. [EChecker.Register\(\)](#); realigns and normalizes 3rd source image. [EChecker.Learn\(RmsDeviation\)](#); processes 3rd image for deviation info. [EChecker.Learn\(Ready\)](#); computes the threshold images.

EChecker.Load

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Load(  
    Euresys.Open_eVision_2_10.ESerializer serializer  
)
```

Parameters

serializer

-

EChecker.Low

Low threshold image for the adaptive segmentation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EImageBW8 Low  
  
{ get; }
```

EChecker.Normalize

Current normalization mode.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.ENormalizationMode Normalize  
  
{ get; set; }
```

EChecker.NumAverageSamples

Number of samples that were accumulated in the "average" phase of the training.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
uint NumAverageSamples
{ get; }
```

EChecker.NumDeviationSamples

Number of samples that were accumulated in the "deviation" phase of the training.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint NumDeviationSamples
{ get; }
```

EChecker.PanX

Current horizontal panning factor for use in display operations.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float PanX
{ get; }
```

EChecker.PanY

Current vertical panning factor for use in display operations.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float PanY  
  
    { get; }
```

EChecker.Register

Realigns and normalizes the source image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Register(  
    )
```

Remarks

Only the inspected ROI is processed. The first time this function is called, the current pattern ROI are used to define the search patterns. After registration, public member [EChecker::Registered](#) contains the realigned, normalized contents of the inspected ROI.

EChecker.Registered

Represents the source image, after it has been aligned with the reference image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EImageBW8 Registered  
    { get; }
```

EChecker.RelativeTolerance

Current tolerance factor to be used for threshold image setup.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float RelativeTolerance  
    { get; set; }
```

EChecker.Save

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Save(  
    Euresys.Open_eVision_2_10.ESerializer serializer  
)
```

Parameters

serializer

-

EChecker.SetPan

Sets the panning factor for use in display operations.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void SetPan(  
    float panX,  
    float panY  
)
```

Parameters

panX

Horizontal panning factor.

panY

Vertical panning factor.

EChecker.SetTolerance

Sets the search margin, i.e. the width and height of the search area surrounding the alignment pattern(s).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void SetTolerance(  
    uint toleranceX,  
    uint toleranceY  
)
```

Parameters

toleranceX

Horizontal search tolerance, in pixels.

toleranceY

Vertical search tolerance, in pixels.

EChecker.SetZoom

Sets the zooming factor for use in display operations.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void SetZoom(  
    float zoom  
)  
  
void SetZoom(  
    float zoomX,  
    float zoomY  
)
```

Parameters

zoom

Magnification factor for zooming in or out in the horizontal and vertical directions (isotropic scaling).

zoomX

Magnification factor for zooming in or out in the horizontal direction.

zoomY

Magnification factor for zooming in or out in the vertical direction.

EChecker.ToleranceX

Current horizontal search tolerance, in pixels.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint ToleranceX  
    { get; }
```

EChecker.ToleranceY

Current vertical search tolerance, in pixels.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint ToleranceY  
    { get; }
```

EChecker.ZoomX

Current horizontal zooming factor for use in display operations.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ZoomX  
  
{ get; }
```

EChecker.ZoomY

Current vertical zooming factor for use in display operations.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ZoomY  
  
{ get; }
```

4.38. ECircle Class

Represents a model of a circle (or arc) in EasyGauge.

Base Class: [EFrame](#)

Namespace: Euresys.Open_eVision_2_10

Properties

[Amplitude](#)

Angular amplitude of the ECircle object.

[Apex](#)

Apex point coordinates of a ECircle object.

ApexAngle

Angular position at the apex of a ECircle object.

ArcLength

Circle arc length of a ECircle object.

Diameter

Diameter of a ECircle object.

Direct

Flag indicating whether positive angles correspond to a clockwise or an anticlockwise rotation.

End

End point coordinates of a ECircle object.

EndAngle

Angular position of the end of a ECircle object.

Full

Flag indicating whether the ECircle object is a full circle or not.

Org

Origin point coordinates of a ECircle object.

OrgAngle

Angular position from where the ECircle object extents.

Radius

M Radius of a ECircle object.

e

thods

CopyTo

Copies all the data of the current [ECircle](#) object into another [ECircle](#) object and returns it.

[Distance](#)

Returns the smallest distance between this [ECircle](#) object and another [ECircle](#).

[ECircle](#)

Constructs a [ECircle](#) object.

[GetDistanceBetweenLineAndCircle](#)

Computes the distance between a line and a circle.

[GetDistanceBetweenPointAndCircle](#)

Computes the distance between a point and a circle.

[GetIntersectionOfCircles](#)

Computes the intersections between two circles. Returns the number of intersections and stores the found intersections in the provided point parameters.

[GetIntersectionOfLineAndCircle](#)

Computes the intersections between a line and circle. Returns the number of intersections and stores the found intersections in the provided point parameters.

[GetPoint](#)

Returns the coordinates of a particular point specified by its location along the circle arc.

[GetProjectionOfPointOnCircle](#)

Computes the projection of a point on a circle.

operator=

Copies all the data from another ECircle object into the current ECircle object

SetFromCenterAndOrigin

Sets the geometric parameters (nominal position and diameter, angular position and amplitude) of an ECircle object.

SetFromOriginMiddleEnd

⌊ Sets the geometric parameters (nominal position and diameter, angular position and amplitude) of an ECircle object.
C

ircle.Amplitude

Angular amplitude of the ECircle object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float Amplitude
```

```
{ get; set; }
```

Remarks

The default value is **360**. A ECircle object is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), the angular position from where it extends, its angular amplitude, and its outline tolerance. The sign of the rotation angle depends whether the field of view is calibrated or not. * When the field of view is calibrated, the coordinate system is said to be direct, the abscissa extends rightwards and the ordinate extends upwards. In this case, an anticlockwise rotation leads to a positive angle value. * When the field of view is not calibrated, the coordinate system is said to be inverse, the abscissa extends rightwards and the ordinate extends downwards. In this case, a clockwise rotation leads to a positive angle value.

ECircle.Apex

Apex point coordinates of a ECircle object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint Apex  
    { get; }
```

ECircle.ApexAngle

Angular position at the apex of a ECircle object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ApexAngle  
    { get; }
```

Remarks

A ECircle object is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), the angular position from where it extends, its angular amplitude, and its outline tolerance. The sign of the rotation angle depends whether the field of view is calibrated or not. * When the field of view is calibrated, the coordinate system is said to be direct, the abscissa extends rightwards and the ordinate extends upwards. In this case, an anticlockwise rotation leads to a positive angle value. * When the field of view is not calibrated, the coordinate system is said to be inverse, the abscissa extends rightwards and the ordinate extends downwards. In this case, a clockwise rotation leads to a positive angle value.

ECircle.ArcLength

Circle arc length of a ECircle object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float ArcLength
{ get; }
```

Remarks

A ECircle object is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), the angular position from where it extents, its angular amplitude, and its outline tolerance.

ECircle.CopyTo

Copies all the data of the current [ECircle](#) object into another [ECircle](#) object and returns it.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ECircle CopyTo(
    Euresys.Open_eVision_2_10.ECircle other
)
```

Parameters

other

Pointer to the [ECircle](#) object in which the current [ECircle](#) object data have to be copied.

Remarks

In case of a **NULL** pointer, a new [ECircle](#) object will be created and returned.

ECircle.Diameter

Diameter of a ECircle object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Diameter  
  
{ get; set; }
```

Remarks

A ECircle object is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), the angular position from where it extents, its angular amplitude, and its outline tolerance. By default, the diameter is **100**, which means 100 pixels when the field of view is not calibrated, and 100 physical units in case of a calibrated field of view.

ECircle.Direct

Flag indicating whether positive angles correspond to a clockwise or an anticlockwise rotation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool Direct  
  
{ get; }
```

Remarks

TRUE (default) means that angles increase anticlockwisely in a direct coordinate system, and clockwisely in an inverse coordinate system. * When the field of view is calibrated, the coordinate system is said to be direct, the abscissa extends rightwards and the ordinate extends upwards. * When the field of view is not calibrated, the coordinate system is said to be inverse, the abscissa extends rightwards and the ordinate extends downwards.

ECircle.Distance

Returns the smallest distance between this ECircle object and another [ECircle](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float Distance(
    Euresys.Open_eVision_2_10.ECircle circle
)
```

Parameters

circle

The other circle

ECircle.ECircle

Constructs a ECircle object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ECircle(
)

void ECircle(
    Euresys.Open_eVision_2_10.EPoint center,
    float diameter,
    float originAngle,
    bool direct
)
```

```

void ECircle(
    Euresys.Open_eVision_2_10.EPoint center,
    Euresys.Open_eVision_2_10.EPoint origin,
    bool direct
)

void ECircle(
    Euresys.Open_eVision_2_10.EPoint center,
    float diameter,
    float originAngle,
    float amplitude
)

void ECircle(
    Euresys.Open_eVision_2_10.EPoint origin,
    Euresys.Open_eVision_2_10.EPoint middle,
    Euresys.Open_eVision_2_10.EPoint end,
    bool fullCircle
)

void ECircle(
    Euresys.Open_eVision_2_10.ECircle other
)

```

Parameters

center

Center coordinates of the circle at its nominal position. The default value is **(0,0)**.

diameter

Nominal diameter of the circle. The default value is **100**.

originAngle

Nominal angular origin of the circle. The default value is 0.

direct

TRUE (default) means that angles increase anticlockwisely in a direct coordinate system.

origin

Origin point coordinates of the circle.

amplitude

Nominal angular amplitude of the circle. The default value is **360**.

middle

Middle point coordinates of the circle.

end

End point coordinates of the circle.

fullCircle

TRUE (default) in case of a full turn circle. If **fullCircle** is **FALSE**, **origin** and **end** give the circle's amplitude.

other

Another ECircle object to be copied in the new ECircle object.

ECircle.End

End point coordinates of a ECircle object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint End  
{ get; }
```

ECircle.EndAngle

Angular position of the end of a ECircle object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float EndAngle  
{ get; }
```


Remarks

A `ECircle` object is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), the angular position from where it extends, its angular amplitude, and its outline tolerance. The sign of the rotation angle depends whether the field of view is calibrated or not. * When the field of view is calibrated, the coordinate system is said to be direct, the abscissa extends rightwards and the ordinate extends upwards. In this case, an anticlockwise rotation leads to a positive angle value. * When the field of view is not calibrated, the coordinate system is said to be inverse, the abscissa extends rightwards and the ordinate extends downwards. In this case, a clockwise rotation leads to a positive angle value.

ECircle.Full

Flag indicating whether the `ECircle` object is a full circle or not.

Namespace: `Euresys.Open_eVision_2_10`

```
[C#]  
  
bool Full  
  
{ get; }
```

Remarks

By default (**TRUE**), the `ECircle` object is a full circle.

ECircle.GetDistanceBetweenLineAndCircle

Computes the distance between a line and a circle.

Namespace: `Euresys.Open_eVision_2_10`

```
[C#]
```

```
float GetDistanceBetweenLineAndCircle(  
    Euresys.Open_eVision_2_10.ELine line,  
    Euresys.Open_eVision_2_10.ECircle circle,  
    bool limited  
)
```

Parameters

- line*
The line.
- circle*
The circle.
- limited*
Indicates if the line and circle parameters should be considered as infinite lines and full circles or as a segments and arcs.

ECircle.GetDistanceBetweenPointAndCircle

Computes the distance between a point and a circle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float GetDistanceBetweenPointAndCircle(  
    Euresys.Open_eVision_2_10.EPoint pt,  
    Euresys.Open_eVision_2_10.ECircle circle,  
    bool limited  
)
```

Parameters

- pt*
The point.
- circle*
The circle.
- limited*

Indicates if the circle parameter should be considered as a full circle or as an arc.

ECircle.GetIntersectionOfCircles

Computes the intersections between two circles. Returns the number of intersections and stores the found intersections in the provided point parameters.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int GetIntersectionOfCircles (
    Euresys.Open_eVision_2_10.ECircle circle1,
    Euresys.Open_eVision_2_10.ECircle circle2,
    Euresys.Open_eVision_2_10.EPoint intersection1,
    Euresys.Open_eVision_2_10.EPoint intersection2,
    bool limited
)
```

Parameters

circle1

The first circle

circle2

The second circle

intersection1

The first intersection

intersection2

The second intersection

limited

Indicates if the circle parameters should be considered as full circles or as arcs.

Remarks

The function returns the number of intersections found. It will return -1 if the two circles are overlapping.

ECircle.GetIntersectionOfLineAndCircle

Computes the intersections between a line and circle. Returns the number of intersections and stores the found intersections in the provided point parameters.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int GetIntersectionOfLineAndCircle(
    Euresys.Open_eVision_2_10.ELine line,
    Euresys.Open_eVision_2_10.ECircle circle,
    Euresys.Open_eVision_2_10.EPoint intersection1,
    Euresys.Open_eVision_2_10.EPoint intersection2,
    bool limited
)
```

Parameters

- line*
The line
- circle*
The circle
- intersection1*
The first intersection
- intersection2*
The second intersection
- limited*
Indicates if the line and circle parameters should be considered as infinite lines and full circles or as a segments and arcs.

Remarks

The function returns the number of intersections found.

ECircle.GetPoint

Returns the coordinates of a particular point specified by its location along the circle arc.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint GetPoint(  
    float fraction  
)
```

Parameters

fraction

Point location expressed as a fraction of the circle arc (range **[-1, +1]**).

ECircle.GetProjectionOfPointOnCircle

Computes the projection of a point on a circle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint GetProjectionOfPointOnCircle(  
    Euresys.Open_eVision_2_10.EPoint pt,  
    Euresys.Open_eVision_2_10.ECircle circle  
)
```

Parameters

pt

The point.

circle

The circle.

ECircle.operator=

Copies all the data from another ECircle object into the current ECircle object

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.ECircle operator=(  
    Euresys.Open_eVision_2_10.ECircle other  
)
```

Parameters

other

ECircle object to be copied

ECircle.Org

Origin point coordinates of a ECircle object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint Org  
{ get; }
```

ECircle.OrgAngle

Angular position from where the ECircle object extents.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float OrgAngle  
  
    { get; }
```

Remarks

A ECircle object is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), the angular position from where it extents, its angular amplitude, and its outline tolerance. The sign of the rotation angle depends whether the field of view is calibrated or not. * When the field of view is calibrated, the coordinate system is said to be direct, the abscissa extends rightwards and the ordinate extends upwards. In this case, an anticlockwise rotation leads to a positive angle value. * When the field of view is not calibrated, the coordinate system is said to be inverse, the abscissa extends rightwards and the ordinate extends downwards. In this case, a clockwise rotation leads to a positive angle value.

ECircle.Radius

Radius of a ECircle object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Radius  
  
    { get; set; }
```

Remarks

A `ECircle` object is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), the angular position from where it extends, its angular amplitude, and its outline tolerance. By default, the radius is **50**, which means 50 pixels when the field of view is not calibrated, and 50 physical units in case of a calibrated field of view.

`ECircle.SetFromCenterAndOrigin`

Sets the geometric parameters (nominal position and diameter, angular position and amplitude) of an `ECircle` object.

Namespace: `Euresys.Open_eVision_2_10`

```
[C#]
void SetFromCenterAndOrigin(
    Euresys.Open_eVision_2_10.EPoint center,
    Euresys.Open_eVision_2_10.EPoint origin,
    bool direct
)
```

Parameters

center

Center coordinates of the circle at its nominal position. The default value is **(0,0)**.

origin

Origin point coordinates of the circle.

direct

TRUE (default) means that angles increase anticlockwisely in a direct coordinate system.

`ECircle.SetFromOriginMiddleEnd`

Sets the geometric parameters (nominal position and diameter, angular position and amplitude) of an `ECircle` object.

Namespace: `Euresys.Open_eVision_2_10`


```
[C#]
void SetFromOriginMiddleEnd(
    Euresys.Open_eVision_2_10.EPoint origin,
    Euresys.Open_eVision_2_10.EPoint middle,
    Euresys.Open_eVision_2_10.EPoint end,
    bool fullCircle
)
```

Parameters

origin

Origin point coordinates of the circle.

middle

Middle point coordinates of the circle.

end

End point coordinates of the circle.

fullCircle

TRUE (default) in case of a full turn circle. If **fullCircle** is **FALSE**, **origin** and **end** give the circle's amplitude.

4.39. ECircleGauge Class

Manages a circle fitting gauge.

Base Class: [ECircleShape](#)

Namespace: Euresys.Open_eVision_2_10

Properties

[Active](#)

Sets the flag indicating whether the gauge is active or not.

[AverageDistance](#)

Average distance between the sampled points and the fitted model.

Circle	Sets the nominal position (center coordinates), diameter, angular origin and amplitude of the circle fitting gauge according to a known circle.
FilteringThreshold	Relative filtering threshold, that is the fraction of the average distance between the sampled points and the fitted model above which a point is filtered out.
HVConstraint	Status of the restriction on the orientation of the point location gauge or model fitting sample paths.
InnerFilteringEnabled	Getter method for the GetInnerFilteringEnabled property. This property is the flag indicating if the inner sampled point filtering is enabled (TRUE), or not.
InnerFilteringThreshold	Sampled point inner filtering threshold.
MeasuredCircle	Information pertaining to the fitted circle.
MinAmplitude	-
MinArea	-
NumFilteringPasses	Number of filtering passes for a model fitting operation.

NumMeasuredPoints	Number of edge-crossing points along the sample path of the gauge that was inspected with the last call to ECircleGauge::MeasureSample .
NumSamples	Number of sampled points during the model fitting operation.
NumSkipRanges	Number of skip ranges in the gauge after a call to ECircleGauge::AddSkipRange .
NumValidSamples	Number of valid sample points remaining after a model fitting operation.
RectangularSamplingArea	-
SamplingStep	Approximate distance between sampled points during a model fitting operation.
Smoothing	Number of pixels used for the low-pass filtering operation.
Thickness	Number of parallel segments used to extract the data profile.
Threshold	-
Tolerance	Searching area half thickness of the circle fitting gauge.

TransitionChoice

-

TransitionIndex

-

TransitionType

-

Type

Shape type.

Valid

M Flag indicating if at least one valid transition has been found.

e

thods

AddSkipRange

Adds an item to the set of skip ranges and returns the index of the newly added range.

CopyTo

Copies all the data of the current ECircleGauge object into another ECircleGauge object, and returns it.

DisableInnerFiltering

Disables inner sampled point filtering.

Drag

Moves a handle to a new position and updates the position parameters of the gauge.

Draw	Draws a graphical representation of a point location or model fitting gauge, as defined by EDrawingMode .
DrawWithCurrentPen	Draws a graphical representation of a point location or model fitting gauge, as defined by EDrawingMode .
ECircleGauge	Constructs a circle measurement context.
GetMeasuredPeak	Returns information pertaining to the default derivative peak, along one of the sample paths of the gauge.
GetMeasuredPoint	Returns the coordinates of a sample point, measured along one of the sample paths of the gauge.
GetMinNumFitSamples	Returns the minimum number of samples required for fitting on each side of the shape.
GetSample	Allows to retrieve the sample points found along the circle.
GetSkipRange	Allows to retrieve the start and end values of the skip range corresponding to the given index, if it is valid (i.e. if it has previously been created by a call to the ECircleGauge::AddSkipRange method).
HitTest	Checks whether the cursor is positioned over a handle (TRUE) or not (FALSE).

Measure	Triggers the point location or the model fitting operation.
MeasureSample	Computes the sample points along the sample path whose index in the list is given by the pathIndex parameter.
MeasureWithoutFitting	Triggers the point location without circle fitting operation.
operator=	Copies all the data from another ECircleGauge object into the current ECircleGauge object
Plot	Draws the profile that is crossed by one of the sample paths of the gauge, as defined by EPlotItem .
PlotWithCurrentPen	Draws the profile that is crossed by one of the sample paths of the gauge, as defined by EPlotItem .
Process	Triggers the process pertaining to a shape or gauge and all the daughter gauges attach to it.
RemoveAllSkipRanges	Removes all the skip ranges previously created by a call to ECircleGauge::AddSkipRange .
RemoveSkipRange	After a call to ECircleGauge::AddSkipRange , removes the skip range with the given index.
SetMinNumFitSamples	Sets the minimum number of samples required for fitting on each side of the shape.

ECircleGauge.Active

Sets the flag indicating whether the gauge is active or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
override bool Active
    { get; set; }
```

Remarks

When complex gauging is required, several gauges can be grouped together. Applying [ECircleGauge::Process](#) to the mother gauge or shape triggers the measurement of the whole. Only the active gauges will participate in the process. By default, the gauge is active (**TRUE**).

ECircleGauge.AddSkipRange

Adds an item to the set of skip ranges and returns the index of the newly added range.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint AddSkipRange(
    uint start,
    uint end
)
```

Parameters

start

Beginning of the skip range.

end

End of the skip range.

Remarks

The samples indices between start and end (including the boundaries) will be discarded during the measurement process.

The `ECircleGauge::AddSkipRange` method allows to define skip ranges in an `ECircleGauge`. This means that, at measure time, samples belonging to these ranges will not be taken into account.

A sample may belong to more than one skip range; to be discarded, a sample has to pertain to at least one range.

Moreover, the skip ranges are allowed to overlap one another.

The range is allowed to be reversed (i.e. end is not required to be greater than start).

Also, start and end are not required to reference valid indices at the time of the call (i.e. the range may lie outside of the current return value for `ECircleGauge::NumSamples`).

ECircleGauge.AverageDistance

Average distance between the sampled points and the fitted model.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float AverageDistance  
{ get; }
```

Remarks

Irrelevant in case of a point location operation.

ECircleGauge.Circle

Sets the nominal position (center coordinates), diameter, angular origin and amplitude of the circle fitting gauge according to a known circle.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
override Euresys.Open_eVision_2_10.ECircle Circle
{ get; set; }
```

ECircleGauge.CopyTo

Copies all the data of the current ECircleGauge object into another ECircleGauge object, and returns it.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ECircleGauge CopyTo(
    Euresys.Open_eVision_2_10.ECircleGauge other,
    bool recursive
)
```

Parameters

other

Pointer to the ECircleGauge object in which the current ECircleGauge object data have to be copied.

recursive

TRUE if the children gauges have to be copied as well, **FALSE** otherwise.

Remarks

In case of a **NULL** pointer, a new ECircleGauge object will be created and returned.

ECircleGauge.DisableInnerFiltering

Disables inner sampled point filtering.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DisableInnerFiltering(
)
```

Remarks

The inner sampled point filtering is activated as soon as the corresponding [ECircleGauge::InnerFilteringThreshold](#) is set.

ECircleGauge.Drag

Moves a handle to a new position and updates the position parameters of the gauge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Drag(
    int x,
    int y
)
```

Parameters

x
Cursor current X coordinate.

y
Cursor current Y coordinate.

ECircleGauge.Draw

Draws a graphical representation of a point location or model fitting gauge, as defined by [EDrawingMode](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingMode

Indicates how the point location or model fitting gauge must be displayed, as defined by [EDrawingMode](#).

daughters

TRUE if the daughters gauges are to be displayed also.

color

The color in which to draw the overlay.

ECircleGauge.DrawWithCurrentPen

Draws a graphical representation of a point location or model fitting gauge, as defined by [EDrawingMode](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawWithCurrentPen (
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingMode

Indicates how the point location or model fitting gauge must be displayed, as defined by [EDrawingMode](#).

daughters

TRUE if the daughters gauges are to be displayed also.

ECircleGauge.ECircleGauge

Constructs a circle measurement context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ECircleGauge (
)

void ECircleGauge (
    Euresys.Open_eVision_2_10.ECircleGauge other
)
```

Parameters

other

Another ECircleGauge object to be copied in the new ECircleGauge object.

Remarks

With the default constructor, all the parameters are initialized to their respective default values. With the copy constructor, the circle measurement context is based on a pre-existing `ECircleGauge` object. The gauge children are also copied. Hierarchy copying through a copy constructor is always recursive. To disable this recursion, use instead the `ECircleGauge::CopyTo` method.

ECircleGauge.FilteringThreshold

Relative filtering threshold, that is the fraction of the average distance between the sampled points and the fitted model above which a point is filtered out.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
float FilteringThreshold  
  
{ get; set; }
```

Remarks

Irrelevant in case of a point location operation.

During a model fitting operation, the "filtering" process can be invoked to remove outliers, i.e. points that were located significantly far away from the fitted model (so that their position is dubious).

ECircleGauge.GetMeasuredPeak

Returns information pertaining to the default derivative peak, along one of the sample paths of the gauge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EPeak GetMeasuredPeak(  
    uint index  
)
```

Parameters

index

This argument must be left unchanged from its default value, i.e. **~0** (= **0xFFFFFFFF**).

Remarks

[ECircleGauge::GetMeasuredPeak](#) returns the information about the derivative peak that meets the following two requirements: 1. It lies on the sample path inspected with the last call to [ECircleGauge::MeasureSample](#), and 1. It is associated with the edge-crossing point along the latter sample path that is selected by the transition choice parameter (cf. [ECircleGauge::TransitionChoice](#)).

Note. For this method to succeed, it is necessary to previously call [ECircleGauge::MeasureSample](#).

ECircleGauge.GetMeasuredPoint

Returns the coordinates of a sample point, measured along one of the sample paths of the gauge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint GetMeasuredPoint(  
    uint index  
)
```

Parameters

index

This argument must be left unchanged from its default value, i.e. **~0** (= **0xFFFFFFFF**).

Remarks

These coordinates pertain to the world space; they are expressed in the reference frame to which the current `ECircleGauge` object belongs. The gauging process uses a list of sample points to find the shape position and size that best fit a given image. These sample points are measured along the sample paths defined by the gauge geometry. `ECircleGauge::GetMeasuredPoint` returns the coordinates of the sample point that meets the following two requirements: 1. It lies on the sample path inspected with the last call to `ECircleGauge::MeasureSample`, and 1. Among all the sample points along the latter sample path, it is the one selected by the `ECircleGauge::TransitionChoice` property.

Note. For this method to succeed, it is necessary to previously call `ECircleGauge::MeasureSample`.

ECircleGauge.GetMinNumFitSamples

Returns the minimum number of samples required for fitting on each side of the shape.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetMinNumFitSamples (
    out int side0,
    out int side1,
    out int side2,
    out int side3
)
```

Parameters

side0

Minimum number of samples on the top side of the rectangle.

side1

Minimum number of samples on the left side of the rectangle.

side2

Minimum number of samples on the bottom side of the rectangle.

side3

Minimum number of samples on the right side of the rectangle.

Remarks

Irrelevant in case of a point location operation.

ECircleGauge.GetSample

Allows to retrieve the sample points found along the circle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool GetSample (
    Euresys.Open_eVision_2_10.EPoint pt,
    uint index
)

void GetSample (
    Euresys.Open_eVision_2_10.ESamplePoint pt,
    uint index
)
```

Parameters

pt

EPoint structure to receive the position of the sample point.

index

The sample index

Remarks

The method provides the sample point corresponding to the supplied index. The returned value corresponds to the sample validity.

ECircleGauge.GetSkipRange

Allows to retrieve the start and end values of the skip range corresponding to the given index, if it is valid (i.e. if it has previously been created by a call to the [ECircleGauge::AddSkipRange](#) method).

Namespace: Euresys.Open_eVision_2_10


```
[C#]
void GetSkipRange(
    uint index,
    out uint start,
    out uint end
)
```

Parameters

index

Index of the skip range.

start

Beginning of the skip range.

end

End of the skip range.

Remarks

Start is guaranteed to be smaller or equal to end.

ECircleGauge.HitTest

Checks whether the cursor is positioned over a handle (**TRUE**) or not (**FALSE**).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool HitTest(
    bool daughters
)
```

Parameters

daughters

TRUE if the daughters gauges handles have to be considered as well.

ECircleGauge.HVConstraint

Status of the restriction on the orientation of the point location gauge or model fitting sample paths.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool HVConstraint  
  
    { get; set; }
```

Remarks

Sample paths are the point location gauges placed along the model to be fitted.

ECircleGauge.InnerFilteringEnabled

Getter method for the **GetInnerFilteringEnabled** property. This property is the flag indicating if the inner sampled point filtering is enabled (**TRUE**), or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool InnerFilteringEnabled  
  
    { get; }
```

Remarks

The inner sampled point filtering is activated as soon as the corresponding [ECircleGauge::InnerFilteringThreshold](#) is set. To disable inner filtering, use the [ECircleGauge::DisableInnerFiltering](#) method.

ECircleGauge.InnerFilteringThreshold

Sampled point inner filtering threshold.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float InnerFilteringThreshold  
  
    { get; set; }
```

Remarks

If inner filtering is enabled, the sampled points that have been found inside the measured circle are filtered in regard of their distance to it. If this distance is greater than the threshold, the sampled point is set as invalid, and removed from the measure. This distance is in physical units. The inner sampled point filtering is activated as soon as the corresponding threshold is set. To disable inner filtering, use the [ECircleGauge::DisableInnerFiltering](#) method.

ECircleGauge.Measure

Triggers the point location or the model fitting operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Measure (  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage  
)
```

Parameters

sourceImage

Pointer to the source image.

Remarks

When this method is called, and if not enough valid points are detected, then the method returns directly, and the measured gauge is set to the nominal parameters.

ECircleGauge.MeasuredCircle

Information pertaining to the fitted circle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ECircle MeasuredCircle
{ get; }
```

ECircleGauge.MeasureSample

Computes the sample points along the sample path whose index in the list is given by the **pathIndex** parameter.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void MeasureSample (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    uint pathIndex
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

pathIndex

Sample path index.

Remarks

This method stores its results into a temporary variable inside the `ECircleGauge` object.

ECircleGauge.MeasureWithoutFitting

Triggers the point location without circle fitting operation.

Namespace: `Euresys.Open_eVision_2_10`

```
[C#]  
  
void MeasureWithoutFitting(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage  
)
```

Parameters

sourceImage
Source image.

Remarks

This method performs the actual measurement for each transition, but does not perform the circle fitting. This means that individual samples will be available through the `ECircleGauge::GetSample` method, but the gauge position will not be changed. Please note that the filtering will not be performed in this method, since it relies upon the fitting process. The filtering parameters will thus be unused.

ECircleGauge.MinAmplitude

-

Namespace: `Euresys.Open_eVision_2_10`

```
[C#]
```

```
uint MinAmplitude  
{ get; set; }
```

ECircleGauge.MinArea

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint MinArea  
{ get; set; }
```

ECircleGauge.NumFilteringPasses

Number of filtering passes for a model fitting operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint NumFilteringPasses  
{ get; set; }
```

Remarks

Irrelevant in case of a point location operation.

During a filtering pass, the points that are too distant from the model are discarded.

During a model fitting operation, the "filtering" process can be invoked to remove outliers, i.e. points that were located significantly far away from the fitted model (so that their position is dubious).

By default (the number of filtering passes is 0), the outliers rejection process is disabled.

ECircleGauge.NumMeasuredPoints

Number of edge-crossing points along the sample path of the gauge that was inspected with the last call to [ECircleGauge::MeasureSample](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint NumMeasuredPoints
{ get; }
```

Remarks

Note. For this method to succeed, it is necessary to previously call [ECircleGauge::MeasureSample](#).

ECircleGauge.NumSamples

Number of sampled points during the model fitting operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint NumSamples
{ get; }
```

Remarks

Irrelevant in case of a point location operation.

After a model fitting operation, a number of points have been fitted along the model. Among them, some are not reliable because of their **Area** value. Among the remaining ones, some are filtered out (**NumFilteringPasses**, **FilteringThreshold**).

ECircleGauge.NumSkipRanges

Number of skip ranges in the gauge after a call to [ECircleGauge::AddSkipRange](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint NumSkipRanges  
  
{ get; }
```

ECircleGauge.NumValidSamples

Number of valid sample points remaining after a model fitting operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint NumValidSamples  
  
{ get; }
```

Remarks

Irrelevant in case of a point location operation.

After a model fitting operation, a number of points have been fitted along the model. Among them, some are not reliable because of their **Area** value. Among the remaining ones, some are filtered out (**NumFilteringPasses**, **FilteringThreshold**).

ECircleGauge.operator=

Copies all the data from another ECircleGauge object into the current ECircleGauge object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ECircleGauge operator=(
    Euresys.Open_eVision_2_10.ECircleGauge other
)
```

Parameters

other

ECircleGauge object to be copied

ECircleGauge.Plot

Draws the profile that is crossed by one of the sample paths of the gauge, as defined by [EPlotItem](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Plot(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.EPlotItem drawItems,
    float width,
    float height,
    float originX,
    float originY
)
```

```

void Plot(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EPlotItem drawItems,
    float width,
    float height,
    float originX,
    float originY
)

void Plot(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.EPlotItem drawItems,
    float width,
    float height,
    float originX,
    float originY
)

```

Parameters

graphicContext

Handle of the device context on which to draw.

drawItems

Boolean combination of [EPlotItem](#) members, that indicates which items must be displayed.

width

Width of the plot.

height

Height of the plot.

originX

Origin point coordinates of the plot along the X axis.

originY

Origin point coordinates of the plot along the Y axis.

color

The color in which to draw the overlay.

Remarks

The sample path that is taken into considered is the one inspected with the last call to [ECircleGauge::MeasureSample](#).

Note. For this method to succeed, it is necessary to previously call [ECircleGauge::MeasureSample](#).

ECircleGauge.PlotWithCurrentPen

Draws the profile that is crossed by one of the sample paths of the gauge, as defined by [EPlotItem](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void PlotWithCurrentPen(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EPlotItem drawItems,
    float width,
    float height,
    float originX,
    float originY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawItems

Boolean combination of [EPlotItem](#) members, that indicates which items must be displayed.

width

Width of the plot.

height

Height of the plot.

originX

Origin point coordinates of the plot along the X axis.

originY

Origin point coordinates of the plot along the Y axis.

Remarks

The sample path that is taken into considered is the one inspected with the last call to [ECircleGauge::MeasureSample](#).

Note. For this method to succeed, it is necessary to previously call [ECircleGauge::MeasureSample](#).

ECircleGauge.Process

Triggers the process pertaining to a shape or gauge and all the daughter gauges attach to it.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Process (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    bool daughters
)
```

Parameters

sourceImage

Pointer to the source image.

daughters

Flag indicating whether the daughters shapes inherit of the same behavior.

Remarks

When complex gauging is required, several gauges can be grouped together. Applying **Process** to the mother gauge or shape triggers the measurement of the whole. Only the active gauges will participate in the process.

ECircleGauge.RectangularSamplingArea

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool RectangularSamplingArea
{ get; set; }
```

ECircleGauge.RemoveAllSkipRanges

Removes all the skip ranges previously created by a call to [ECircleGauge::AddSkipRange](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void RemoveAllSkipRanges (  
)
```

ECircleGauge.RemoveSkipRange

After a call to [ECircleGauge::AddSkipRange](#), removes the skip range with the given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void RemoveSkipRange (  
    uint index  
)
```

Parameters

index

Index of the skip range to remove, as returned by [ECircleGauge::AddSkipRange](#).

ECircleGauge.SamplingStep

Approximate distance between sampled points during a model fitting operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float SamplingStep
{ get; set; }
```

Remarks

Irrelevant in case of a point location operation.

To fit a model, a series of point location operations are performed along the model. The point location gauges spacing is given by the sampling step.

By default, the sampling step is set to **5**, which means 5 pixels when the field of view is not calibrated, and 5 physical units in case of a calibrated field of view.

Be aware that if the sampling step is too large, the number of sampled points along the model will not be sufficient enough to accurately locate it.

ECircleGauge.SetMinNumFitSamples

Sets the minimum number of samples required for fitting on each side of the shape.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetMinNumFitSamples (
    int side0,
    int side1,
    int side2,
    int side3
)
```

Parameters

side0

Required number of samples to correctly fit the circle. The default value is **3**. It is the only parameter taken into account.

side1

Not used.

side2

Not used.

side3

Not used.

Remarks

Irrelevant in case of a point location operation. When the [ECircleGauge::Measure](#) method is called, and if not enough valid points are detected, then the method returns directly, and the measured gauge is set to the nominal parameters.

ECircleGauge.Smoothing

Number of pixels used for the low-pass filtering operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
uint Smoothing  
  
{ get; set; }
```

Remarks

To reduce the effect of noise, the profile data can be low-pass filtered along the point location gauge direction.

ECircleGauge.Thickness

Number of parallel segments used to extract the data profile.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint Thickness
```

```
{ get; set; }
```

Remarks

To reduce the effect of noise and/or strengthen a transition, several parallel profiles can be accumulated.

ECircleGauge.Threshold

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint Threshold
```

```
{ get; set; }
```

ECircleGauge.Tolerance

Searching area half thickness of the circle fitting gauge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float Tolerance
```

```
{ get; set; }
```


Remarks

A circle fitting gauge is fully defined knowing its nominal position (given by the coordinates of its center), its nominal radius (diameter), the angular position from where it extends, its angular amplitude and its outline tolerance. By default, the searching area thickness of the circle fitting gauge is **20** (2x10), which means 20 pixels when the field of view is not calibrated, and 20 physical units in case of a calibrated field of view.

ECircleGauge.TransitionChoice

-

Namespace: Euresys.Open_eVision_2_10

[C#]

Euresys.Open_eVision_2_10.ETransitionChoice TransitionChoice

{ get; set; }

ECircleGauge.TransitionIndex

-

Namespace: Euresys.Open_eVision_2_10

[C#]

uint TransitionIndex

{ get; set; }

ECircleGauge.TransitionType

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.ETransitionType TransitionType  
  
    { get; set; }
```

ECircleGauge.Type

Shape type.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
override Euresys.Open_eVision_2_10.EShapeType Type  
  
    { get; }
```

ECircleGauge.Valid

Flag indicating if at least one valid transition has been found.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool Valid  
  
    { get; }
```

Remarks

A **FALSE** value means that no measurement has been performed. A **TRUE** value means that a transition was found along the sample path inspected with the last call to [ECircleGauge::MeasureSample](#), and thus a point has been measured.

4.40. ECircleRegion Class

Manages a complete context for an [ERegion](#) shaped like a circle.

Base Class: [ERegion](#)

Namespace: Euresys.Open_eVision_2_10

Properties

Center

Center of the region

Radius

M Radius of the region

e

Methods

Drag

Moves the specified handle to a new position and updates all placement parameters of the region.

ECircleRegion

Constructs an [ECircleRegion](#) context.

HitTest

Detects if the cursor is placed over one of the dragging handles.

Load

Loads the [ECircleRegion](#). The given [ESerializer](#) must have been created for reading.

operator!=

Checks if this [ECircleRegion](#) instance is not strictly equal to another

operator=

Assignment operator.

operator==

Checks if this [ECircleRegion](#) instance is strictly equal to another

Save

Saves the [ECircleRegion](#). The given [ESerializer](#) must have been created for writing.

Scale

Creates a new region by scaling the [ECircleRegion](#).

Translate

Creates a new [ECircleRegion](#) by translating the [ECircleRegion](#).

C

ircleRegion.Center

Center of the region

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EPoint Center  
  
{ get; set; }
```

ECircleRegion.Drag

Moves the specified handle to a new position and updates all placement parameters of the region.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void Drag(  
    int x,  
    int y,  
    float zoomX,  
    float zoomY,  
    int panX,  
    int panY  
)
```

Parameters

x

x-coordinate of the mouse cursor.

y

y-coordinate of the mouse cursor.

zoomX

Horizontal zoom factor. By default, true scale is used.

zoomY

Vertical zoom factor. By default, true scale is used.

panX

Horizontal pan offset. By default, no pan is added.

panY

Vertical pan offset. By default, no pan is added.

Remarks

If zooming and/or panning were used when drawing the region, the same values must be used with [ECircleRegion::HitTest](#) and [ECircleRegion::Drag](#).

ECircleRegion.ECircleRegion

Constructs an [ECircleRegion](#) context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ECircleRegion(
)

void ECircleRegion(
    float centerX,
    float centerY,
    float radius
)

void ECircleRegion(
    Euresys.Open_eVision_2_10.EPoint center,
    float radius
)

void ECircleRegion(
    Euresys.Open_eVision_2_10.EPoint pt1,
    Euresys.Open_eVision_2_10.EPoint pt2,
    Euresys.Open_eVision_2_10.EPoint pt3
)

void ECircleRegion(
    Euresys.Open_eVision_2_10.ECircle circle
)

void ECircleRegion(
    Euresys.Open_eVision_2_10.ECircleRegion other
)
```

Parameters

centerX

The abscissa of the center of the [ECircleRegion](#).

centerY

The ordinate of the center of the [ECircleRegion](#).

radius

The radius of the [ECircleRegion](#).

center

The center of the [ECircleRegion](#).

pt1

One of the three points defining the [ECircleRegion](#).

pt2

One of the three points defining the [ECircleRegion](#).

pt3

One of the three points defining the [ECircleRegion](#).

circle

The result of an [ECircleGauge](#) object.

other

[ECircleRegion](#) context to copy.

Remarks

When defining a [ECircleRegion](#), the resulting radius value must not be **0** else an [EError](#) is thrown.

When defining a [ECircleRegion](#), the resulting radius value must be small enough so that the region fit in memory.

When defining a [ECircleRegion](#) with three points, they must be non aligned else an [EError](#) is thrown.

ECircleRegion.HitTest

Detects if the cursor is placed over one of the dragging handles.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
Euresys.Open_eVision_2_10.EEditionMode HitTest(  
    int x,  
    int y,  
    float zoomX,  
    float zoomY,  
    int panX,  
    int panY  
)
```

Parameters

x

x-coordinate of the mouse cursor.

y

y-coordinate of the mouse cursor.

zoomX

Horizontal zoom factor. By default, true scale is used.

zoomY

Vertical zoom factor. By default, true scale is used.

panX

Horizontal pan offset. By default, no pan is added.

panY

Vertical pan offset. By default, no pan is added.

Remarks

Returns a handle identifier, as defined by [EEditionMode](#).

If zooming and/or panning were used when drawing the region, the same values must be used with [ECircleRegion::HitTest](#) and [ECircleRegion::Drag](#).

ECircleRegion.Load

Loads the [ECircleRegion](#). The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
void Load(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The serializer.

ECircleRegion.operator!=

Checks if this [ECircleRegion](#) instance is not strictly equal to another

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool operator!=(
    Euresys.Open_eVision_2_10.ECircleRegion other
)
```

Parameters

other

Reference to the other [ECircleRegion](#) instance

ECircleRegion.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ECircleRegion operator=(
    Euresys.Open_eVision_2_10.ECircleRegion other
)
```

Parameters

other

Reference to the [ECircleRegion](#) used for the assignment

ECircleRegion.operator==

Checks if this [ECircleRegion](#) instance is strictly equal to another

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool operator==(
    Euresys.Open_eVision_2_10.ECircleRegion other
)
```

Parameters

other

Reference to the other [ECircleRegion](#) instance

ECircleRegion.Radius

Radius of the region

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float Radius
{ get; set; }
```

ECircleRegion.Save

Saves the [ECircleRegion](#). The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer
The serializer.

ECircleRegion.Scale

Creates a new region by scaling the [ECircleRegion](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.ECircleRegion Scale(  
    float scale  
)  
  
Euresys.Open_eVision_2_10.EEllipseRegion Scale(  
    float scaleX,  
    float scaleY  
)
```

Parameters

scale

Isotropic scale.

scaleX

Horizontal scale.

scaleY

Vertical scale.

ECircleRegion.Translate

Creates a new [ECircleRegion](#) by translating the [ECircleRegion](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
Euresys.Open_eVision_2_10.ECircleRegion Translate(  
    float dx,  
    float dy  
)
```

Parameters

dx

Horizontal translation in pixel value

dy

Vertical translation in pixel value

4.41. ECircleShape Class

Base Class: [EShape](#)

Derived Class(es): [ECircleGauge](#)

Namespace: Euresys.Open_eVision_2_10

Properties

Amplitude	Angular amplitude of the ECircleShape object.
Angle	-
Apex	Apex point coordinates of a ECircleShape object.
ApexAngle	Angular position at the apex of a ECircleShape object.
ArcLength	Circle arc length of a ECircleShape object.
Center	-
CenterX	Abscissa of the origin point of the frame.
CenterY	Ordinate of the origin point of the frame.

Circle	-
Diameter	-
Direct	Flag indicating whether positive angles correspond to a clockwise or an anticlockwise rotation.
End	End point coordinates of a ECircleShape object.
EndAngle	Angular position of the end of a ECircleShape object.
Full	Flag indicating whether the ECircleShape object is a full circle or not.
Org	Origin point coordinates of a ECircleShape object.
OrgAngle	Angular position from where the ECircleShape object extents.
Radius	-
Scale	-
Type	Shape type.

Methods

Closest

Find the daughter shape that is the closest to this shape. To retrieve the closest shape, use [EShape::ClosestShape](#).

CopyTo

Copies all the data of the current `ECircleShape` object into another `ECircleShape` object, and returns it.

Drag

Moves a handle to a new position and updates the position parameters of the gauge.

Draw

Draws a graphical representation of a shape, as defined by [EDrawingMode](#).

DrawWithCurrentPen

Draws a graphical representation of a shape, as defined by [EDrawingMode](#).

GetPoint

Returns the coordinates of a particular point specified by its location along the circle arc.

HitTest

Checks if there is a handle under the cursor.

operator=

Copies all the data from another `ECircleShape` object into the current `ECircleShape` object

SetCenterXY

Sets the center coordinates of a [ECircleShape](#) object.

SetFromCenterAndOrigin

Sets the geometric parameters (nominal position and diameter, angular position and amplitude) of an [ECircleShape](#) object.

SetFromOriginMiddleEnd

⌈ Sets the geometric parameters (nominal position and diameter, angular position and amplitude) of an [ECircleShape](#) object.
⌋
C

ircleShape.Amplitude

Angular amplitude of the [ECircleShape](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Amplitude  
{ get; set; }
```

Remarks

The default value is **360**. A [ECircleShape](#) object is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), the angular position from where it extents, its angular amplitude, and its outline tolerance. The sign of the rotation angle depends whether the field of view is calibrated or not. * When the field of view is calibrated, the coordinate system is said to be direct, the abscissa extends rightwards and the ordinate extends upwards. In this case, an anticlockwise rotation leads to a positive angle value. * When the field of view is not calibrated, the coordinate system is said to be inverse, the abscissa extends rightwards and the ordinate extends downwards. In this case, a clockwise rotation leads to a positive angle value.

ECircleShape.Angle

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Angle  
    { get; set; }
```

ECircleShape.Apex

Apex point coordinates of a ECircleShape object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint Apex  
    { get; }
```

ECircleShape.ApexAngle

Angular position at the apex of a ECircleShape object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float ApexAngle
{ get; }
```

Remarks

A `ECircleShape` object is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), the angular position from where it extends, its angular amplitude, and its outline tolerance. The sign of the rotation angle depends whether the field of view is calibrated or not. * When the field of view is calibrated, the coordinate system is said to be direct, the abscissa extends rightwards and the ordinate extends upwards. In this case, an anticlockwise rotation leads to a positive angle value. * When the field of view is not calibrated, the coordinate system is said to be inverse, the abscissa extends rightwards and the ordinate extends downwards. In this case, a clockwise rotation leads to a positive angle value.

ECircleShape.ArcLength

Circle arc length of a `ECircleShape` object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float ArcLength
{ get; }
```

Remarks

A `ECircleShape` object is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), the angular position from where it extends, its angular amplitude, and its outline tolerance.

ECircleShape.Center

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint Center  
  
{ get; set; }
```

ECircleShape.CenterX

Abscissa of the origin point of the frame.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float CenterX  
  
{ get; }
```

ECircleShape.CenterY

Ordinate of the origin point of the frame.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float CenterY  
  
{ get; }
```

ECircleShape.Circle

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
virtual Euresys.Open_eVision_2_10.ECircle Circle  
{ get; set; }
```

ECircleShape.Closest

Find the daughter shape that is the closest to this shape. To retrieve the closest shape, use [EShape::ClosestShape](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Closest(  
)
```

ECircleShape.CopyTo

Copies all the data of the current ECircleShape object into another ECircleShape object, and returns it.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ECircleShape CopyTo(
    Euresys.Open_eVision_2_10.ECircleShape dest,
    bool bRecursive
)
```

Parameters

dest
-
bRecursive
-

Remarks

In case of a **NULL** pointer, a new ECircleShape object will be created and returned.

ECircleShape.Diameter

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float Diameter
{ get; set; }
```

ECircleShape.Direct

Flag indicating whether positive angles correspond to a clockwise or an anticlockwise rotation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool Direct
{ get; }
```

Remarks

TRUE (default) means that angles increase anticlockwisely in a direct coordinate system, and clockwisely in an inverse coordinate system. * When the field of view is calibrated, the coordinate system is said to be direct, the abscissa extends rightwards and the ordinate extends upwards. * When the field of view is not calibrated, the coordinate system is said to be inverse, the abscissa extends rightwards and the ordinate extends downwards.

ECircleShape.Drag

Moves a handle to a new position and updates the position parameters of the gauge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Drag(
    int n32CursorX,
    int n32CursorY
)
```

Parameters

n32CursorX
-
n32CursorY
-

ECircleShape.Draw

Draws a graphical representation of a shape, as defined by [EDrawingMode](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingMode

Indicates how the point location or model fitting gauge must be displayed, as defined by [EDrawingMode](#).

daughters

TRUE if the daughters gauges are to be displayed also.

color

-

ECircleShape.DrawWithCurrentPen

Draws a graphical representation of a shape, as defined by [EDrawingMode](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawWithCurrentPen(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingMode

Indicates how the point location or model fitting gauge must be displayed, as defined by [EDrawingMode](#).

daughters

TRUE if the daughters gauges are to be displayed also.

ECircleShape.End

End point coordinates of a ECircleShape object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint End
{ get; }
```

ECircleShape.EndAngle

Angular position of the end of a ECircleShape object.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
float EndAngle
{ get; }
```

Remarks

A `ECircleShape` object is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), the angular position from where it extends, its angular amplitude, and its outline tolerance. The sign of the rotation angle depends whether the field of view is calibrated or not. * When the field of view is calibrated, the coordinate system is said to be direct, the abscissa extends rightwards and the ordinate extends upwards. In this case, an anticlockwise rotation leads to a positive angle value. * When the field of view is not calibrated, the coordinate system is said to be inverse, the abscissa extends rightwards and the ordinate extends downwards. In this case, a clockwise rotation leads to a positive angle value.

ECircleShape.Full

Flag indicating whether the `ECircleShape` object is a full circle or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool Full
{ get; }
```

Remarks

By default (**TRUE**), the `ECircleShape` object is a full circle.

ECircleShape.GetPoint

Returns the coordinates of a particular point specified by its location along the circle arc.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint GetPoint(
    float fraction
)
```

Parameters

fraction

Point location expressed as a fraction of the circle arc (range **[-1, +1]**).

ECircleShape.HitTest

Checks if there is a handle under the cursor.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool HitTest(
    bool bDaughters
)
```

Parameters

bDaughters

Indicates if the check must be done in the whole hierarchy or just this object.

ECircleShape.operator=

Copies all the data from another ECircleShape object into the current ECircleShape object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ECircleShape operator=(
    Euresys.Open_eVision_2_10.ECircleShape other
)
```

Parameters

other

ECircleShape object to be copied

ECircleShape.Org

Origin point coordinates of a ECircleShape object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint Org
{ get; }
```

ECircleShape.OrgAngle

Angular position from where the ECircleShape object extents.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float OrgAngle
{ get; }
```

Remarks

A `ECircleShape` object is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), the angular position from where it extends, its angular amplitude, and its outline tolerance. The sign of the rotation angle depends whether the field of view is calibrated or not. * When the field of view is calibrated, the coordinate system is said to be direct, the abscissa extends rightwards and the ordinate extends upwards. In this case, an anticlockwise rotation leads to a positive angle value. * When the field of view is not calibrated, the coordinate system is said to be inverse, the abscissa extends rightwards and the ordinate extends downwards. In this case, a clockwise rotation leads to a positive angle value.

ECircleShape.Radius

-

Namespace: Euresys.Open_eVision_2_10

[C#]

float Radius

{ get; set; }

ECircleShape.Scale

-

Namespace: Euresys.Open_eVision_2_10

[C#]

float Scale

{ get; set; }

ECircleShape.SetCenterXY

Sets the center coordinates of a [ECircleShape](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetCenterXY(
    float centerX,
    float centerY
)
```

Parameters

centerX

Center coordinates of the [ECircleShape](#) object.

centerY

Center coordinates of the [ECircleShape](#) object.

ECircleShape.SetFromCenterAndOrigin

Sets the geometric parameters (nominal position and diameter, angular position and amplitude) of an [ECircleShape](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetFromCenterAndOrigin(
    Euresys.Open_eVision_2_10.EPoint center,
    Euresys.Open_eVision_2_10.EPoint origin,
    bool direct
)
```

Parameters

center

Center coordinates of the circle at its nominal position. The default value is **(0,0)**.

origin

Origin point coordinates of the circle.

direct

TRUE (default) means that angles increase anticlockwisely in a direct coordinate system.

ECircleShape.SetFromOriginMiddleEnd

Sets the geometric parameters (nominal position and diameter, angular position and amplitude) of an ECircleShape object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetFromOriginMiddleEnd(
    Euresys.Open_eVision_2_10.EPoint origin,
    Euresys.Open_eVision_2_10.EPoint middle,
    Euresys.Open_eVision_2_10.EPoint end,
    bool fullCircle
)
```

Parameters

origin

Origin point coordinates of the circle.

middle

Middle point coordinates of the circle.

end

End point coordinates of the circle.

fullCircle

TRUE (default) in case of a full turn circle. If **fullCircle** is **FALSE**, **origin** and **end** give the circle's amplitude.

ECircleShape.Type

Shape type.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
override Euresys.Open_eVision_2_10.EShapeType Type  
{ get; }
```

4.42. EClassificationDataset Class

[EClassificationDataset](#) manages a dataset of labeled images to be used for classification.

The dataset must contain at least two different labels and each label can be assigned to any number of image in the dataset.

Labels are represented by strings. Images can be given to a [EClassificationDataset](#) as a string containing the path to a stored image or using a supported Open eVision image structure. Supported structures are 8-bits monochrome ([EImageBW8](#)), 16-bits monochrome ([EImageBW16](#)), and 24-bits color ([EROIC24](#), [EImageC24](#)).

A [EClassificationDataset](#) object is also responsible for providing tools to do data augmentation.

Data augmentation is the process of generating new images on-the-fly by applying affine transformations to those already in the dataset. Data augmentation allows a deep neural network to be invariant to the applied transformation without having to capture and label real world images containing those transformations.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

Properties

[Channels](#)

Number of channels of the first image added to the dataset.

[EnableDataAugmentation](#)

Enable data augmentation.

EnableHorizontalFlip

Enable horizontal flipping in data augmentation.

EnableVerticalFlip

Enable vertical flipping in data augmentation.

GaussianNoiseMaximumStandardDeviation

Sets the Gaussian noise maximum standard deviation. The Gaussian noise is an additive noise sampled from a Gaussian distribution of deviation between [EClassificationDataset::GaussianNoiseMinimumStandardDeviation](#) and [EClassificationDataset::GaussianNoiseMaximumStandardDeviation](#) (also called the normal distribution). Its value must be superior or equal to [EClassificationDataset::GaussianNoiseMinimumStandardDeviation](#).

GaussianNoiseMinimumStandardDeviation

Sets the Gaussian noise minimum standard deviation. The Gaussian noise is an additive noise sampled from a Gaussian distribution of deviation between [EClassificationDataset::GaussianNoiseMinimumStandardDeviation](#) and [EClassificationDataset::GaussianNoiseMaximumStandardDeviation](#) (also called the normal distribution). Its value must be between **0** and [EClassificationDataset::GaussianNoiseMaximumStandardDeviation](#).

Height

Height of the first image added to the dataset.

MaxBrightnessOffset

Maximum absolute brightness offset. Its value must be between **0** and **1**.

MaxContrastGain

Maximum contrast gain. Its value must be strictly positive and over [EClassificationDataset::MinContrastGain](#).

MaxGamma

Maximum gamma for gamma correction. Its value must be higher than `EClassificationDataset::MinGamma`.

MaxHorizontalShear

Maximum absolute horizontal shear. It is represented as an angle from the vertical direction. Its value must be between **0** and **90°**.

MaxHorizontalShift

Maximum horizontal shift for data augmentation. The horizontal shift will be between `-EClassificationDataset::MaxHorizontalShift` and `+EClassificationDataset::MaxHorizontalShift`.

MaxHueOffset

Maximum absolute hue offset. Its value must be between **0** and **180°**.

MaxRotationAngle

Maximum rotation angle for data augmentation. The rotation angle will be between `-EClassificationDataset::MaxRotationAngle` and `+EClassificationDataset::MaxRotationAngle`.

MaxSaturationGain

Maximum saturation gain. Its value must be over or equal to `EClassificationDataset::MinSaturationGain`.

MaxScale

Maximum scaling allowed for data augmentation.

MaxVerticalShear

Maximum absolute vertical shear. It is represented as an angle from the horizontal direction. Its value must be between **0** and **90°**.

MaxVerticalShift

Maximum vertical shift for data augmentation. The vertical shift will be between `-EClassificationDataset::MaxHorizontalShift` and `+EClassificationDataset::MaxHorizontalShift`.

MinContrastGain

Minimum contrast gain. Its value must be strictly positive and below `EClassificationDataset::MaxContrastGain`.

MinGamma

Minimum gamma for gamma correction. Its value must be strictly positive and below `EClassificationDataset::MaxGamma`.

MinSaturationGain

Minimum saturation gain. Its value must be strictly positive.

MinScale

Minimum scaling allowed for data augmentation.

NumImages

Number of images in the dataset.

NumLabels

Number of labels in the dataset.

SaltAndPepperNoiseMaximumDensity

Sets the maximum density of the salt and pepper noise. The salt and pepper noise sets the value of a number of randomly selected (between `EClassificationDataset::SaltAndPepperNoiseMinimumDensity` and `EClassificationDataset::SaltAndPepperNoiseMaximumDensity`) pixels to its minimum or maximum value. Its value must be between `EClassificationDataset::SaltAndPepperNoiseMinimumDensity` and **100**.

SaltAndPepperNoiseMinimumDensity

Sets the minimum density of the salt and pepper noise. The salt and pepper noise sets the value of a number of randomly selected (between `EClassificationDataset::SaltAndPepperNoiseMinimumDensity` and `EClassificationDataset::SaltAndPepperNoiseMaximumDensity`) pixels to its minimum or maximum value. Its value must be between `0` and `EClassificationDataset::SaltAndPepperNoiseMaximumDensity`.

SpeckleNoiseMaximumStandardDeviation

Sets the speckle noise maximum standard deviation. The speckle noise is a multiplicative noise sampled from a Gamma distribution with a mean of 1 and with its standard deviation between `EClassificationDataset::SpeckleNoiseMinimumStandardDeviation` and `EClassificationDataset::SpeckleNoiseMaximumStandardDeviation`. Its value must be strictly higher than `EClassificationDataset::SpeckleNoiseMinimumStandardDeviation`.

SpeckleNoiseMinimumStandardDeviation

Sets the speckle noise minimum standard deviation. The speckle noise is a multiplicative noise sampled from a Gamma distribution with a mean of 1 and with its standard deviation between `EClassificationDataset::SpeckleNoiseMinimumStandardDeviation` and `EClassificationDataset::SpeckleNoiseMaximumStandardDeviation`. Its value must be strictly positive and lower than `EClassificationDataset::SpeckleNoiseMaximumStandardDeviation`.

Width

M Width of the first image added to the dataset.

thods

e

AddImage

Adds an image with its label to the dataset. The image can be specified by its path on the filesystem (parameter `imagePath`) or by an Open eVision image buffer (parameter `img`).

The method returns **-1** if there was an error when inserting the image in the dataset or a numeric identifier greater or equal to **0** that can be used to access and manipulate the image in the dataset.

AddImages

Adds all the images present in the directory specified by the parameter `path` and whose filename matches the filter and associates them to the corresponding label.

The method returns the number of images added to the dataset.

Clear

Clears the datasets.

EClassificationDataset

Constructs a [EClassificationDataset](#) object.

Export

Exports the dataset and its images to the given directory. An export will create a sub-directory for each label and export the images of the dataset to their corresponding label sub-directories in the PNG format.

Finally, a new [EClassificationDataset](#) object with relative paths to the images is saved into the given directory.

GetImage

Gets a copy of the `i`-th image of the dataset. The user is responsible for clearing the memory of the returned pointer.

GetImageLabel

Gets the label of the `i`-th image of the dataset.

GetImagePath

Gets the path of the i-th image of the dataset. If the image was not given using a path, the method will throw an exception.

GetImages

Gets a copy of all images in the dataset. If data augmentation is enabled, the returned images will be augmented versions of the ones in the dataset.

The caller is responsible for clearing the memory allocated for each image.

GetImagesIndexesWithLabel

Gets a list of index corresponding to the images associated with the given label

GetLabel

Gets the i-th label of the dataset (starting from **0** to `EClassificationDataset::NumLabels - 1`)

GetLabelWeight

Gets the weight associated to the i-th label of the dataset (starting from **0** to `EClassificationDataset::NumLabels - 1`)

Load

Loads a classification dataset from disk.

operator=

Assignment operator

Save

Saves a classification dataset to disk, containing the file paths to the images in the dataset and their associated label.

Serialize

Serializes the dataset state, which is the file paths to the images in the dataset and their associated label.

SetImageLabel

Sets the label of images in the dataset.

SetLabel

Sets the i-th label of the dataset (starting from **0** to `EClassificationDataset::NumLabels - 1`). This operation does not add a new label to the dataset but simply renames an existing label.

SetLabelWeight

Sets the weight associated to the i-th label of the dataset (starting from **0** to `EClassificationDataset::NumLabels - 1`). This operation does not add a new label.

SplitDataset

⌊ Splits the dataset in two parts to be used for training and validation respectively.

C

lassificationDataset.AddImage

Adds an image with its label to the dataset. The image can be specified by its path on the filesystem (parameter `imagePath`) or by an Open eVision image buffer (parameter `img`).

The method returns **-1** if there was an error when inserting the image in the dataset or a numeric identifier greater or equal to **0** that can be used to access and manipulate the image in the dataset.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

[C#]

```
int AddImage(  
    string imagePath,  
    string label  
)  
  
int AddImage(  
    Euresys.Open_eVision_2_10.EBaseROI img,  
    string label  
)
```

Parameters

imagePath

The path to an image

label

The label

img

The image

EClassificationDataset.AddImages

Adds all the images present in the directory specified by the parameter path and whose filename matches the filter and associates them to the corresponding label.

The method returns the number of images added to the dataset.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
  
int AddImages(  
    string filter,  
    string label  
)
```

Parameters

filter

A glob filter

label

A label.

Remarks

The filter is a glob pattern. This means the wildcard characters "*" and "?" correspond to "zero or more character" and "a single character" respectively. For example, the filter "*_good_*.png" will match any filename that contains the string "_good_" and has a png extension.

EClassificationDataset.Channels

Number of channels of the first image added to the dataset.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
uint Channels  
    { get; }
```

EClassificationDataset.Clear

Clears the datasets.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
void Clear(  
    )
```


EClassificationDataset.EClassificationDataset

Constructs a [EClassificationDataset](#) object.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
void EClassificationDataset(
)
void EClassificationDataset(
    Euresys.Open_eVision_2_10.EasyDeepLearning.EClassificationDataset other
)
```

Parameters

other

Reference to the [EClassificationDataset](#) object that should be copied

EClassificationDataset.EnableDataAugmentation

Enable data augmentation.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
bool EnableDataAugmentation
{ get; set; }
```

EClassificationDataset.EnableHorizontalFlip

Enable horizontal flipping in data augmentation.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
bool EnableHorizontalFlip
    { get; set; }
```

EClassificationDataset.EnableVerticalFlip

Enable vertical flipping in data augmentation.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
bool EnableVerticalFlip
    { get; set; }
```

EClassificationDataset.Export

Exports the dataset and its images to the given directory. An export will create a sub-directory for each label and export the images of the dataset to their corresponding label sub-directories in the PNG format. Finally, a new [EClassificationDataset](#) object with relative paths to the images is saved into the given directory.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
void Export(
    string directory
)
```

Parameters

directory

A string containing the full path to the directory.

EClassificationDataset.GaussianNoiseMaximumStandardDeviation

Sets the Gaussian noise maximum standard deviation.

The Gaussian noise is an additive noise sampled from a Gaussian distribution of deviation between

[EClassificationDataset::GaussianNoiseMinimumStandardDeviation](#) and

[EClassificationDataset::GaussianNoiseMaximumStandardDeviation](#) (also called the normal distribution).

Its value must be superior or equal to [EClassificationDataset::GaussianNoiseMinimumStandardDeviation](#).

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
float GaussianNoiseMaximumStandardDeviation
{
    get; set;
}
```

Remarks

This noise is computed before the salt and paper noise.

EClassificationDataset.GaussianNoiseMinimumStandardDeviation

Sets the Gaussian noise minimum standard deviation.

The Gaussian noise is an additive noise sampled from a Gaussian distribution of deviation between [EClassificationDataset::GaussianNoiseMinimumStandardDeviation](#) and [EClassificationDataset::GaussianNoiseMaximumStandardDeviation](#) (also called the normal distribution). Its value must be between **0** and [EClassificationDataset::GaussianNoiseMaximumStandardDeviation](#).

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
float GaussianNoiseMinimumStandardDeviation
{ get; set; }
```

Remarks

This noise is computed before the salt and paper noise.

EClassificationDataset.GetImage

Gets a copy of the i-th image of the dataset. The user is responsible for clearing the memory of the returned pointer.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
Euresys.Open_eVision_2_10.EBaseROI GetImage (
    int i
)
```

Parameters

i

The index of the image.

EClassificationDataset.GetImageLabel

Gets the label of the *i*-th image of the dataset.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
string GetImageLabel (
    int i
)
```

Parameters

i

The index of the image for which to get the label.

EClassificationDataset.GetImagePath

Gets the path of the *i*-th image of the dataset. If the image was not given using a path, the method will throw an exception.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
string GetImagePath (
    int i
)
```

Parameters

i

The index of the image for which to get the path.

EClassificationDataset.GetImages

Gets a copy of all images in the dataset. If data augmentation is enabled, the returned images will be augmented versions of the ones in the dataset.

The caller is responsible for clearing the memory allocated for each image.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
Euresys.Open_eVision_2_10.EBaseROI[] GetImages (
)
Euresys.Open_eVision_2_10.EBaseROI[] GetImages (
    string label
)
```

Parameters

label

The label.

EClassificationDataset.GetImagesIndexesWithLabel

Gets a list of index corresponding to the images associated with the given label

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
```

```
uint[] GetImagesIndexesWithLabel (
    string label
)

uint[] GetImagesIndexesWithLabel (
    int labelIndex
)
```

Parameters

label

The label

labelIndex

The index of the label (starting from **0** to [EClassificationDataset::NumLabels - 1](#))

EClassificationDataset.GetLabel

Gets the i-th label of the dataset (starting from **0** to [EClassificationDataset::NumLabels - 1](#))

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
string GetLabel (
    int i
)
```

Parameters

i

Label index

EClassificationDataset.GetLabelWeight

Gets the weight associated to the i-th label of the dataset (starting from **0** to [EClassificationDataset::NumLabels - 1](#))

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
float GetLabelWeight(  
    int i  
)
```

Parameters

i
Label index

EClassificationDataset.Height

Height of the first image added to the dataset.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
uint Height  
{ get; }
```

EClassificationDataset.Load

Loads a classification dataset from disk.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning


```
[C#]
void Load(
    string path
)
```

Parameters

path

A string containing the full path to the dataset file.

EClassificationDataset.MaxBrightnessOffset

Maximum absolute brightness offset. Its value must be between **0** and **1**.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
float MaxBrightnessOffset
{ get; set; }
```

Remarks

Brightness transformation is performed by adding a value taken between [-EClassificationDataset::MaxBrightnessOffset](#) and [+EClassificationDataset::MaxBrightnessOffset](#) to each pixel of the normalized image.

EClassificationDataset.MaxContrastGain

Maximum contrast gain. Its value must be strictly positive and over [EClassificationDataset::MinContrastGain](#).

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
float MaxContrastGain
{ get; set; }
```

Remarks

Contrast transformation is performed by multiplying each mean-centered pixel value by a gain value taken between [EClassificationDataset::MinContrastGain](#) and [EClassificationDataset::MaxContrastGain](#). A contrast transformation does not change the overall brightness of an image.

EClassificationDataset.MaxGamma

Maximum gamma for gamma correction. Its value must be higher than [EClassificationDataset::MinGamma](#).

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
float MaxGamma
{ get; set; }
```

Remarks

Gamma correction transformation is performed by raising each normalized pixel value to the power of gamma with gamma taken between [EClassificationDataset::MinGamma](#) and [EClassificationDataset::MaxGamma](#).

EClassificationDataset.MaxHorizontalShear

Maximum absolute horizontal shear.
It is represented as an angle from the vertical direction. Its value must be between **0** and **90°**.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
```

```
float MaxHorizontalShear  
  
{ get; set; }
```

EClassificationDataset.MaxHorizontalShift

Maximum horizontal shift for data augmentation.
The horizontal shift will be between [-EClassificationDataset::MaxHorizontalShift](#) and [+EClassificationDataset::MaxHorizontalShift](#).

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
```

```
int MaxHorizontalShift  
  
{ get; set; }
```

EClassificationDataset.MaxHueOffset

Maximum absolute hue offset. Its value must be between **0** and **180°**.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
```

```
float MaxHueOffset  
  
{ get; set; }
```

Remarks

The hue is represented as an angle between **0** and **360°**. The hue transformation is performed by rotating the hue of each pixel by a value between **-EClassificationDataset::MaxHueOffset** and **+EClassificationDataset::MaxHueOffset**. This transformation only works for color images.

EClassificationDataset.MaxRotationAngle

Maximum rotation angle for data augmentation.

The rotation angle will be between **-EClassificationDataset::MaxRotationAngle** and **+EClassificationDataset::MaxRotationAngle**.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
float MaxRotationAngle
{ get; set; }
```

EClassificationDataset.MaxSaturationGain

Maximum saturation gain. Its value must be over or equal to **EClassificationDataset::MinSaturationGain**.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
float MaxSaturationGain
{ get; set; }
```

Remarks

The saturation transformation is performed by multiplying the saturation of each pixel by a value between **EClassificationDataset::MinSaturationGain** and **EClassificationDataset::MaxSaturationGain**.

EClassificationDataset.MaxScale

Maximum scaling allowed for data augmentation.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
float MaxScale  
  
{ get; set; }
```

EClassificationDataset.MaxVerticalShear

Maximum absolute vertical shear.

It is represented as an angle from the horizontal direction. Its value must be between **0** and **90°**.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
float MaxVerticalShear  
  
{ get; set; }
```

EClassificationDataset.MaxVerticalShift

Maximum vertical shift for data augmentation.

The vertical shift will be between **-EClassificationDataset::MaxHorizontalShift** and **+EClassificationDataset::MaxHorizontalShift**.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
int MaxVerticalShift
{ get; set; }
```

EClassificationDataset.MinContrastGain

Minimum contrast gain. Its value must be strictly positive and below [EClassificationDataset::MaxContrastGain](#).

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
float MinContrastGain
{ get; set; }
```

Remarks

Contrast transformation is performed by multiplying each mean-centered pixel value by a gain value taken between [EClassificationDataset::MinContrastGain](#) and [EClassificationDataset::MaxContrastGain](#). A contrast transformation does not change the overall brightness of an image.

EClassificationDataset.MinGamma

Minimum gamma for gamma correction. Its value must be strictly positive and below [EClassificationDataset::MaxGamma](#).

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
```

```
float MinGamma
```

```
{ get; set; }
```

Remarks

Gamma correction transformation is performed by raising each normalized pixel value to the power of gamma with gamma taken between [EClassificationDataset::MinGamma](#) and [EClassificationDataset::MaxGamma](#).

EClassificationDataset.MinSaturationGain

Minimum saturation gain. Its value must be strictly positive.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
```

```
float MinSaturationGain
```

```
{ get; set; }
```

Remarks

The saturation transformation is performed by multiplying the saturation of each pixel by a value between [EClassificationDataset::MinSaturationGain](#) and [EClassificationDataset::MaxSaturationGain](#).

EClassificationDataset.MinScale

Minimum scaling allowed for data augmentation.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
```

```
float MinScale
```

```
{ get; set; }
```

EClassificationDataset.NumImages

Number of images in the dataset.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
```

```
int NumImages
```

```
{ get; }
```

EClassificationDataset.NumLabels

Number of labels in the dataset.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
```

```
int NumLabels
```

```
{ get; }
```


EClassificationDataset.operator=

Assignment operator

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
Euresys.Open_eVision_2_10.EasyDeepLearning.EClassificationDataset operator=(  
    Euresys.Open_eVision_2_10.EasyDeepLearning.EClassificationDataset other  
)
```

Parameters

other

Reference to the [EClassificationDataset](#) object used for the assignment

EClassificationDataset.SaltAndPepperNoiseMaximumDensity

Sets the maximum density of the salt and pepper noise.

The salt and pepper noise sets the value of a number of randomly selected (between

[EClassificationDataset::SaltAndPepperNoiseMinimumDensity](#) and

[EClassificationDataset::SaltAndPepperNoiseMaximumDensity](#)) pixels to its minimum or maximum value.

Its value must be between [EClassificationDataset::SaltAndPepperNoiseMinimumDensity](#) and **100**.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
  
float SaltAndPepperNoiseMaximumDensity  
  
    { get; set; }
```

Remarks

This noise is computed after all the other noises.

EClassificationDataset.SaltAndPepperNoiseMinimumDensity

Sets the minimum density of the salt and pepper noise.

The salt and pepper noise sets the value of a number of randomly selected (between

[EClassificationDataset::SaltAndPepperNoiseMinimumDensity](#) and

[EClassificationDataset::SaltAndPepperNoiseMaximumDensity](#)) pixels to its minimum or maximum value.

Its value must be between **0** and [EClassificationDataset::SaltAndPepperNoiseMaximumDensity](#).

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
float SaltAndPepperNoiseMinimumDensity
{ get; set; }
```

Remarks

This noise is computed after all the other noises.

EClassificationDataset.Save

Saves a classification dataset to disk, containing the file paths to the images in the dataset and their associated label.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
void Save(
    string path
)
```

Parameters

path

A string containing the full path to the dataset file.

Remarks

This method only save the image that were given to this [EClassificationDataset](#) instance as [EBaseROI](#) pointers. To obtain a portable [EClassificationDataset](#) file, please use [EClassificationDataset::Export](#).

EClassificationDataset.Serialize

Serializes the dataset state, which is the file paths to the images in the dataset and their associated label.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
void Serialize(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

Pointer to the [ESerializer](#) object that is read from or written to.

EClassificationDataset.SetImageLabel

Sets the label of images in the dataset.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
void SetImageLabel(
    int i,
    string label
)
```

```
void SetImageLabel(  
    string filter,  
    string label  
)
```

Parameters

i

The index of the image for which to set the label.

label

The label

filter

A glob filter

Remarks

The filter is a glob pattern. This means the wildcard characters "*" and "?" correspond to "zero or more character" and "a single character" respectively. For example, the filter "*_good_*.png" will match any filename that contains the string "_good_" and has a png extension.

EClassificationDataset.SetLabel

Sets the *i*-th label of the dataset (starting from **0** to [EClassificationDataset::NumLabels - 1](#)). This operation does not add a new label to the dataset but simply renames an existing label.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
  
void SetLabel(  
    int i,  
    string label  
)
```

Parameters

i

Label index

label

Replacement label

EClassificationDataset.SetLabelWeight

Sets the weight associated to the *i*-th label of the dataset (starting from **0** to [EClassificationDataset::NumLabels - 1](#)). This operation does not add a new label.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
void SetLabelWeight(
    int i,
    float weight
)
```

Parameters

i

Label index

weight

-

EClassificationDataset.SpeckleNoiseMaximumStandardDeviation

Sets the speckle noise maximum standard deviation.

The speckle noise is a multiplicative noise sampled from a Gamma distribution with a mean of 1 and with its standard deviation between [EClassificationDataset::SpeckleNoiseMinimumStandardDeviation](#) and [EClassificationDataset::SpeckleNoiseMaximumStandardDeviation](#).

Its value must be strictly higher than [EClassificationDataset::SpeckleNoiseMinimumStandardDeviation](#).

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
float SpeckleNoiseMaximumStandardDeviation
{ get; set; }
```

Remarks

This noise is computed before the salt and paper noise.

EClassificationDataset.SpeckleNoiseMinimumStandardDeviation

Sets the speckle noise minimum standard deviation.

The speckle noise is a multiplicative noise sampled from a Gamma distribution with a mean of 1 and with its standard deviation between [EClassificationDataset::SpeckleNoiseMinimumStandardDeviation](#) and [EClassificationDataset::SpeckleNoiseMaximumStandardDeviation](#).

Its value must be strictly positive and lower than [EClassificationDataset::SpeckleNoiseMaximumStandardDeviation](#).

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
float SpeckleNoiseMinimumStandardDeviation
{ get; set; }
```

Remarks

This noise is computed before the salt and paper noise.

EClassificationDataset.SplitDataset

Splits the dataset in two parts to be used for training and validation respectively.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
void SplitDataset(
    Euresys.Open_eVision_2_10.EasyDeepLearning.EClassificationDataset d1,
    Euresys.Open_eVision_2_10.EasyDeepLearning.EClassificationDataset d2,
    float proportion,
    bool random
)
```

Parameters

d1

First part of the dataset

d2

Second part of the dataset

proportion

Proportion of image of each class to put into the first part. The remaining images are put in *d2*

random

Randomly sample the images.

EClassificationDataset.Width

Width of the first image added to the dataset.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
uint Width
{ get; }
```

4.43. EClassificationMetrics Class

Collection of metrics used to evaluate the state of an [EClassifier](#).

A metric is a value summarizing a collection of classification results ([EClassificationResult](#)). New results can be added to the object individually with [EClassificationMetrics::AddResult](#) or collectively with [EClassificationMetrics::AddMetrics](#).

[EClassificationMetrics](#) contains the following metrics:

- the accuracy (see [EClassificationMetrics::Accuracy](#)) - the error (see [EClassificationMetrics::Error](#))

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

Properties

Accuracy

The accuracy of the classifier.

The accuracy is the number of images that were correctly classified (also called the true positives) over the total number of images that was used to evaluate the classifier.

Error

MThe error of the classifier.

The error, which is also called the loss, is the quantity that is minimized during the training of the deep neural network. For classification, the error is the crossentropy.

Methods

AddMetrics

Adds the other metrics to the current metrics of this object.

AddResult

Adds the given result with the corresponding groundtruth label to the metrics.

EClassificationMetrics

Constructs an [EClassificationMetrics](#) object.

GetConfusion

Confusion value of one label with another.
The confusion value of a label with another is the number of images belonging to this label that are classified as belonging to the other label.

IsValid

Indicates whether the object contains at least one classification result.

Load

Loads a classification metric. The given [ESerializer](#) must have been created for reading.

operator=

Assignment operator

Save

Saves a classification metric. The given [ESerializer](#) must have been created for writing.

Serialize

Serializes the metrics.

C

lassificationMetrics.Accuracy

The accuracy of the classifier.

The accuracy is the number of images that were correctly classified (also called the true positives) over the total number of images that was used to evaluate the classifier.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
float Accuracy
{ get; }
```

EClassificationMetrics.AddMetrics

Adds the other metrics to the current metrics of this object.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
void AddMetrics (
    Euresys.Open_eVision_2_10.EasyDeepLearning.EClassificationMetrics other
)
```

Parameters

other
Classification metrics

EClassificationMetrics.AddResult

Adds the given result with the corresponding groundtruth label to the metrics.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
```

```
void AddResult(  
    Euresys.Open_eVision_2_10.EasyDeepLearning.EClassificationResult result,  
    string groundtruthLabel  
)
```

Parameters

result

A reference to an [EClassificationResult](#) object.

groundtruthLabel

The groundtruth label corresponding to the result

EClassificationMetrics.EClassificationMetrics

Constructs an [EClassificationMetrics](#) object.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
void EClassificationMetrics(  
    )  
  
void EClassificationMetrics(  
    Euresys.Open_eVision_2_10.EasyDeepLearning.EClassificationMetrics other  
    )
```

Parameters

other

Reference to the [EClassificationMetrics](#) object that should be copied

EClassificationMetrics.Error

The error of the classifier.

The error, which is also called the loss, is the quantity that is minimized during the training of the deep neural network. For classification, the error is the crossentropy.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
  
float Error  
  
{ get; }
```

EClassificationMetrics.GetConfusion

Confusion value of one label with another.

The confusion value of a label with another is the number of images belonging to this label that are classified as belonging to the other label.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
  
uint GetConfusion(  
    string trueClass,  
    string predictedClass  
)
```

Parameters

trueClass

The label for which to obtain the confusion value

predictedClass

The label with which there is a confusion

EClassificationMetrics.IsValid

Indicates whether the object contains at least one classification result.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
bool IsValid(  
    )
```

EClassificationMetrics.Load

Loads a classification metric. The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
void Load(  
    Euresys.Open_eVision_2_10.ESerializer serializer  
    )
```

Parameters

serializer

Pointer to [ESerializer](#) created for reading.

EClassificationMetrics.operator=

Assignment operator

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
Euresys.Open_eVision_2_10.EasyDeepLearning.EClassificationMetrics operator=(
    Euresys.Open_eVision_2_10.EasyDeepLearning.EClassificationMetrics other
)
```

Parameters

other

Reference to the [EClassificationMetrics](#) object used for the assignment

EClassificationMetrics.Save

Saves a classification metric. The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

Pointer to [ESerializer](#) created for writing.

EClassificationMetrics.Serialize

Serializes the metrics.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
void Serialize(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

Pointer to [ESerializer](#)

4.44. EClassificationResult Class

An [EClassificationResult](#) object represents the result of a classification.

The most probable label and its probability are accessible through the methods [EClassificationResult::BestLabel](#) and [EClassificationResult::BestProbability](#).

The probability and ranking of all labels are accessible through the [EClassificationResult](#) and [EClassificationResult](#) methods.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

Properties

[BestLabel](#)

Gets the most probable label.

[BestLabelId](#)

Gets the most probable label id.

BestProbability

M Gets the probability associated with the most probable label

e

Methods

EClassificationResult

Constructs a non-valid [EClassificationResult](#).

GetProbability

Gets the probability corresponding to the given label.

GetRanking

Gets the ranking corresponding to the given label. The ranking goes from **1** (most probable label) to [EClassifier::NumLabels](#) (least probable label).

IsValid

Indicates whether the result was produced by [EClassifier](#). A default constructed [EClassificationResult](#) is not valid.

operator=

E Assignment operator

C

EClassificationResult.BestLabel

Gets the most probable label.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

[C#]


```
string BestLabel
```

```
{ get; }
```

EClassificationResult.BestLabelId

Gets the most probable label id.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
```

```
int BestLabelId
```

```
{ get; }
```

EClassificationResult.BestProbability

Gets the probability associated with the most probable label

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
```

```
float BestProbability
```

```
{ get; }
```

EClassificationResult.EClassificationResult

Constructs a non-valid [EClassificationResult](#).

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
void EClassificationResult(
)
void EClassificationResult(
    Euresys.Open_eVision_2_10.EasyDeepLearning.EClassificationResult other
)
```

Parameters

other

Reference to the [EClassificationResult](#) object that should be copied

EClassificationResult.GetProbability

Gets the probability corresponding to the given label.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
float GetProbability(
    string label
)
```

Parameters

label

The label

EClassificationResult.GetRanking

Gets the ranking corresponding to the given label. The ranking goes from **1** (most probable label) to [EClassifier::NumLabels](#) (least probable label).

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
  
int GetRanking(  
    string label  
)
```

Parameters

label
The label

EClassificationResult.IsValid

Indicates whether the result was produced by [EClassifier](#).
A default constructed [EClassificationResult](#) is not valid.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
  
bool IsValid(  
)
```

EClassificationResult.operator=

Assignment operator

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
Euresys.Open_eVision_2_10.EasyDeepLearning.EClassificationResult operator=(
    Euresys.Open_eVision_2_10.EasyDeepLearning.EClassificationResult other
)
```

Parameters

other

Reference to the [EClassificationResult](#) object used for the assignment

4.45. EClassifier Class

[EClassifier](#) allows to train a classifier using an [EClassificationDataset](#) object and classify new images.

As required by Deep Learning techniques, the input image of [EClassifier](#) must be of the same format (width, height, number of channels). By default, this format will be the one of the first image added to the dataset used for training unless its width and height is smaller than the minimum width and height supported by the classifier (See [EClassifier::MinimumWidth](#) and [EClassifier::MinimumHeight](#)). In this case, the input resolution will be the minimum resolution supported by the classifier. The format can also be specified by the [EClassifier::Width](#), [EClassifier::Height](#) and [EClassifier::Channels](#). methods.

By default, images that don't satisfy the image format of the classifier are automatically reformatted. This behavior can be controlled through the [EClassifier::EnableAutomaticImageReformat](#) method. When the automatic image reformatting is disabled, training or classifying an image that doesn't satisfy the input image format will result in an exception.

Once trained, the input image format cannot be changed.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

Properties

BatchSize

Batch size. The batch size is the number of images that are processed together during training and batch classification ([EClassifier::Classify](#)).

When using multi-GPUs processing (see [EClassifier::GPUIndexes](#)), the batch size is the number of images that each GPU will process at once.

A large batch size will increase the processing speed on GPU but also the memory requirements.

The batch size must be bigger or equal to **2** and it is commonly chosen to be a power of 2.

BatchSizeForMaximumClassificationSpeed

Computes the batch size that will maximize the classification speed on a GPU.

BestIteration

Iteration at which the minimum validation error was reached. After training, the classifier is in the state it was at this best iteration.

Channels

Number of channels for input images of the classifier. The number of channels can be either 1 (monochrome image) or 3 (RGB image). By default, this value will be set from the format of the first image added to the training dataset.

CurrentTrainingFinishedIterations

Number of iterations that are already finished in the current training.

CurrentTrainingNumIterations

Total number of training iterations that will be performed during the current training of the classifier.

CurrentTrainingProgression

Current training progression as a percentage (between 0 and 1).

EnableAutomaticImageReformat

Enable automatic image reformat.

EnableGPU

Enable the use of a GPU for training and classification.

EnableHistogramEqualization

Enable histogram equalization of all images passing through the classifier.

GPUIndexes

Indexes of the GPUs to use for computations.
Using multiple GPUs is only possible when we can process multiple images at once, i.e. during training ([EClassifier::Train](#)) or batch classification ([EClassifier::Classify](#)). Thus, when performing a classification for a single image, only the first GPU in the parameter 'indexes' will be used.
By default, all the detected GPUs will be used.

Height

Height for input images of the classifier. By default, this value will be set from the format of the first image of the dataset used for training.

ImageCacheSize

Size in byte of the image cache. The cache is used during training to store reformatted and normalized images. A correctly sized cache can reduce the hard drive accesses and the preprocessing time for each image.

MinimumHeight

Minimum height for input images of the classifier. This value is equal to **128**.

MinimumWidth

Minimum width for input images of the classifier. This value is equal to **128**.

NumGPUs

Number of detected GPU.

NumLabels

Number of labels of this classifier. If the classifier is not trained, the method will throw an exception.

NumTrainedIterations

Number of iterations that were performed to train this classifier.

OptimizeBatchSize

Indicates whether to optimize the batch size (see [EClassifier::BatchSize](#)) at training to maximize the training speed according to [EClassifier::EnableGPU](#) and the available memory.

Width

M Width for input images of the classifier. By default, this value will be set from the format of the first image of the dataset used for **e** training.

thods

Classify

Classifies images and returns the complete results as an [EClassificationResult](#) object.
The method throws an exception if the input image does not fulfill the input specification.

EClassifier

Constructs a [EClassifier](#) object.

Evaluate	Evaluates the EClassifier using the given EClassificationDataset .
GetHeatMap	Gets a heatmap associated with the given label. A heatmap is an image that indicates which pixels in the original image best explain the given label.
GetLabel	Gets the label from its index. If the classifier is not trained, the method will throw an exception.
GetTrainingMetrics	Gets the metrics obtained with the training dataset at the given iteration. The iterations are indexed between 0 and EClassifier::NumTrainedIterations - 1 .
GetValidationMetrics	Gets the metrics obtained with the validation dataset at the given iteration. The iterations are indexed between 0 and EClassifier::NumTrainedIterations - 1 .
IsTrained	Tells whether the classifier has been trained.
IsTraining	Indicates whether the object is currently training.
Load	Loads a classifier. The given ESerializer must have been created for reading.
operator=	Assignment operator

Save	Saves a classifier. The given ESerializer must have been created for writing.
Serialize	Serializes the classifier.
StopTraining	Stops training and returns the last completed iteration. If the parameter 'wait' is set to true, the method will wait for the training thread to completely stop. Otherwise, the method will return immediately.
Train	Trains the EClassifier with the given dataset for the specified number of iterations. At the end of the training, the classifier is in the state it was at the iteration that gave the minimum validation error. See EClassifier::BestIteration .
WaitForIterationCompletion	Waits until an iteration is complete. A call to this method will block the calling thread until a training iteration in the training thread is finished. This method returns the number of trained iterations.
WaitForTrainingCompletion	Waits until the training is complete or the timeout is expired. A call to this method will block the calling thread for the shortest time between the timeout and the time it takes for the training to complete. A negative timeout means that the method will wait until the training is complete. The default value is set to -1 . The method returns the number of trained iterations.

EClassifier.BatchSize

Batch size. The batch size is the number of images that are processed together during training and batch classification ([EClassifier::Classify](#)).

When using multi-GPUs processing (see [EClassifier::GPUIndexes](#)), the batch size is the number of images that each GPU will process at once.

A large batch size will increase the processing speed on GPU but also the memory requirements.

The batch size must be bigger or equal to **2** and it is commonly chosen to be a power of 2.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
int BatchSize
    { get; set; }
```

Remarks

If [EClassifier::OptimizeBatchSize](#) is 'true', then the value of this property can be changed automatically during training. See also [EClassifier::BatchSizeForMaximumClassificationSpeed](#).

EClassifier.BatchSizeForMaximumClassificationSpeed

Computes the batch size that will maximize the classification speed on a GPU.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
int BatchSizeForMaximumClassificationSpeed
    { get; }
```

Remarks

This value is given as an indication and should not necessarily be used in practice.

You must choose a tradeoff between the overall classification speed (also called the throughput), which is limited by this value, and the time it takes to classify a whole batch (also called the latency), which is minimized by making classifications image per image (i.e. a batch size of 1).

The tradeoff depends on your particular application.

EClassifier.BestIteration

Iteration at which the minimum validation error was reached. After training, the classifier is in the state it was at this best iteration.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
int BestIteration
{ get; }
```

EClassifier.Channels

Number of channels for input images of the classifier. The number of channels can be either 1 (monochrome image) or 3 (RGB image). By default, this value will be set from the format of the first image added to the training dataset.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
uint Channels
{ get; set; }
```

Remarks

If the classifier is not trained or the value was not explicitly set, its value will be **0**.

EClassifier.Classify

Classifies images and returns the complete results as an [EClassificationResult](#) object.
The method throws an exception if the input image does not fulfill the input specification.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
Euresys.Open_eVision_2_10.EasyDeepLearning.EClassificationResult Classify(
    Euresys.Open_eVision_2_10.EBaseROI img
)

Euresys.Open_eVision_2_10.EasyDeepLearning.EClassificationResult[] Classify(
    ref Euresys.Open_eVision_2_10.EBaseROI[] imgList
)

Euresys.Open_eVision_2_10.EasyDeepLearning.EClassificationResult[] Classify(
    ref Euresys.Open_eVision_2_10.EImageBW8[] imgList
)

Euresys.Open_eVision_2_10.EasyDeepLearning.EClassificationResult[] Classify(
    ref Euresys.Open_eVision_2_10.EImageBW16[] imgList
)

Euresys.Open_eVision_2_10.EasyDeepLearning.EClassificationResult[] Classify(
    ref Euresys.Open_eVision_2_10.EImageC24[] imgList
)
```

Parameters

img

Image to classify

imgList

Vector of images to classify

Remarks

Classifying a set of images is usually faster than classifying each image sequentially.

To maximize the classification speed on a GPU, [EClassifier::BatchSize](#) and the size of the set of input images must be equal to the value returned by [EClassifier](#).

EClassifier.CurrentTrainingFinishedIterations

Number of iterations that are already finished in the current training.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
uint CurrentTrainingFinishedIterations  
    { get; }
```

EClassifier.CurrentTrainingNumIterations

Total number of training iterations that will be performed during the current training of the classifier.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
uint CurrentTrainingNumIterations  
    { get; }
```

EClassifier.CurrentTrainingProgression

Current training progression as a percentage (between 0 and 1).

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
float CurrentTrainingProgression
{ get; }
```

EClassifier.EClassifier

Constructs a [EClassifier](#) object.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
void EClassifier(
)
void EClassifier(
    Euresys.Open_eVision_2_10.EasyDeepLearning.EClassifier other
)
```

Parameters

other

Reference to the [EClassifier](#) object that should be copied

EClassifier.EnableAutomaticImageReformat

Enable automatic image reformat.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
```

```
bool EnableAutomaticImageReformat  
  
{ get; set; }
```

EClassifier.EnableGPU

Enable the use of a GPU for training and classification.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
```

```
bool EnableGPU  
  
{ get; set; }
```

EClassifier.EnableHistogramEqualization

Enable histogram equalization of all images passing through the classifier.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
```

```
bool EnableHistogramEqualization  
  
{ get; set; }
```

EClassifier.Evaluate

Evaluates the [EClassifier](#) using the given [EClassificationDataset](#).

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
Euresys.Open_eVision_2_10.EasyDeepLearning.EClassificationMetrics Evaluate(
    Euresys.Open_eVision_2_10.EasyDeepLearning.EClassificationDataset dataset
)
```

Parameters

dataset

[EClassificationDataset](#) with which to evaluate the classifier

EClassifier.GetHeatMap

Gets a heatmap associated with the given label. A heatmap is an image that indicates which pixels in the original image best explain the given label.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
Euresys.Open_eVision_2_10.EImageBW8 GetHeatMap(
    Euresys.Open_eVision_2_10.EBaseROI image,
    string label
)
```

Parameters

image

A reference to the image we want to generate the heatmap from.

label

A reference to the string representing the label we want to explain for the given image.

EClassifier.GetLabel

Gets the label from its index. If the classifier is not trained, the method will throw an exception.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
string GetLabel(
    uint index
)
```

Parameters

index
Index of the label

EClassifier.GetTrainingMetrics

Gets the metrics obtained with the training dataset at the given iteration.
The iterations are indexed between **0** and [EClassifier::NumTrainedIterations - 1](#).

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
Euresys.Open_eVision_2_10.EasyDeepLearning.EClassificationMetrics GetTrainingMetrics(
    int id
)
```

Parameters

id

The iteration index

EClassifier.GetValidationMetrics

Gets the metrics obtained with the validation dataset at the given iteration.
The iterations are indexed between **0** and [EClassifier::NumTrainedIterations - 1](#).

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
Euresys.Open_eVision_2_10.EasyDeepLearning.EClassificationMetrics GetValidationMetrics(  
    int id  
)
```

Parameters

id
The iteration index

EClassifier.GPUIndexes

Indexes of the GPUs to use for computations.
Using multiple GPUs is only possible when we can process multiple images at once, i.e. during training ([EClassifier::Train](#)) or batch classification ([EClassifier::Classify](#)). Thus, when performing a classification for a single image, only the first GPU in the parameter 'indexes' will be used.
By default, all the detected GPUs will be used.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
uint[] GPUIndexes
```

```
{ get; set; }
```

Remarks

The GPU are indexed from **0** to `EClassifier::NumGPUs - 1`.

EClassifier.Height

Height for input images of the classifier. By default, this value will be set from the format of the first image of the dataset used for training.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
  
uint Height  
  
{ get; set; }
```

Remarks

If the classifier is not trained or the value was not explicitly set, its value will be **0**.

EClassifier.ImageCacheSize

Size in byte of the image cache. The cache is used during training to store reformatted and normalized images. A correctly sized cache can reduce the hard drive accesses and the preprocessing time for each image.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
  
System.UInt64 ImageCacheSize
```

```
{ get; set; }
```

EClassifier.IsTrained

Tells whether the classifier has been trained.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
bool IsTrained(  
    )
```

Remarks

When the classifier has been trained, the input image format of the classifier is fixed and can be obtained with the methods [EClassifier::Width](#), [EClassifier::Height](#), [EClassifier::Channels](#), and [EClassifier](#).

EClassifier.IsTraining

Indicates whether the object is currently training.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
bool IsTraining(  
    )
```

EClassifier.Load

Loads a classifier. The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
void Load(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

Pointer to the [ESerializer](#) created for reading.

EClassifier.MinimumHeight

Minimum height for input images of the classifier. This value is equal to **128**.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
uint MinimumHeight
{ get; }
```

EClassifier.MinimumWidth

Minimum width for input images of the classifier. This value is equal to **128**.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
uint MinimumWidth  
    { get; }
```

EClassifier.NumGPUs

Number of detected GPU.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
uint NumGPUs  
    { get; }
```

EClassifier.NumLabels

Number of labels of this classifier. If the classifier is not trained, the method will throw an exception.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
uint NumLabels  
    { get; }
```

EClassifier.NumTrainedIterations

Number of iterations that were performed to train this classifier.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
uint NumTrainedIterations
{ get; }
```

Remarks

This number of iteration may result from the addition of the iterations performed in several calls to [EClassifier::Train](#). An iteration can also be called an epoch.

EClassifier.operator=

Assignment operator

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
Euresys.Open_eVision_2_10.EasyDeepLearning.EClassifier operator=(
    Euresys.Open_eVision_2_10.EasyDeepLearning.EClassifier other
)
```

Parameters

other

Reference to the [EClassifier](#) object used for the assignment

EClassifier.OptimizeBatchSize

Indicates whether to optimize the batch size (see [EClassifier::BatchSize](#)) at training to maximize the training speed according to [EClassifier::EnableGPU](#) and the available memory.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
bool OptimizeBatchSize
{ get; set; }
```

EClassifier.Save

Saves a classifier. The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

Pointer to the [ESerializer](#) created for writing.

EClassifier.Serialize

Serializes the classifier.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
void Serialize(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

Pointer to [ESerializer](#)

EClassifier.StopTraining

Stops training and returns the last completed iteration. If the parameter 'wait' is set to true, the method will wait for the training thread to completely stop. Otherwise, the method will return immediately.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
int StopTraining(
    bool wait
)
```

Parameters

wait

Whether to wait for the training to completely stop

EClassifier.Train

Trains the [EClassifier](#) with the given dataset for the specified number of iterations. At the end of the training, the classifier is in the state it was at the iteration that gave the minimum validation error. See [EClassifier::BestIteration](#).

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
void Train(
    Euresys.Open_eVision_2_10.EasyDeepLearning.EClassificationDataset dataset,
    int iterations
)

void Train(
    Euresys.Open_eVision_2_10.EasyDeepLearning.EClassificationDataset trainingDataset,
    Euresys.Open_eVision_2_10.EasyDeepLearning.EClassificationDataset validationDataset,
    int iterations
)
```

Parameters

dataset

[EClassificationDataset](#) with which to train and validate the classifier

iterations

Number of iterations for training.

trainingDataset

[EClassificationDataset](#) with which to train the classifier

validationDataset

[EClassificationDataset](#) with which to validate the classifier

EClassifier.WaitForIterationCompletion

Waits until an iteration is complete. A call to this method will block the calling thread until a training iteration in the training thread is finished. This method returns the number of trained iterations.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
  
int WaitForIterationCompletion(  
    )
```

EClassifier.WaitForTrainingCompletion

Waits until the training is complete or the timeout is expired. A call to this method will block the calling thread for the shortest time between the timeout and the time it takes for the training to complete. A negative timeout means that the method will wait until the training is complete. The default value is set to **-1**. The method returns the number of trained iterations.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]  
  
int WaitForTrainingCompletion(  
    int timeout  
    )
```

Parameters

timeout

Timeout in second

EClassifier.Width

Width for input images of the classifier. By default, this value will be set from the format of the first image of the dataset used for training.

Namespace: Euresys.Open_eVision_2_10.EasyDeepLearning

```
[C#]
```

```
uint Width
```

```
{ get; set; }
```

Remarks

If the classifier is not trained or the value was not explicitly set, its value will be **0**.

4.46. ECodedElement Class

This class encapsulates either an object or a hole in an object, in a coded image.

Remarks

This abstract class provides a large set of methods applicable to a particular coded element. The set includes methods to get the features of a coded element, to draw coded elements, and to render flexible masks.

Derived Class(es): [EObject](#) [EHole](#)

Namespace: Euresys.Open_eVision_2_10

Properties

Area | Returns the number of pixels inside the coded element.

BottomLimit | Returns the highest (integer) Y-coordinate of all the pixels of the coded element.

BoundingBox | Returns the bounding box of the coded element (Ferret box at orientation 0°).

BoundingBoxCenter | Returns the coordinates of the center of the bounding box of the coded element.

BoundingBoxCenterX	Returns the abscissa of the center of the bounding box of the coded element.
BoundingBoxCenterY	Returns the ordinate of the center of the bounding box of the coded element.
BoundingBoxHeight	Returns the height of the bounding box (Ferret diameter at 90°).
BoundingBoxWidth	Returns the width of the bounding box (Ferret diameter at 0°).
Contour	Returns the coordinates of the starting point of the countour of the coded element.
ContourX	Returns the abscissa of the starting point of the countour of the coded element.
ContourY	Returns the ordinate of the starting point of the countour of the coded element.
Eccentricity	Returns the eccentricity of the ellipse of inertia.
ElementIndex	Returns the index of the coded element.
EllipseAngle	Returns the angle of the ellipse of inertia.
EllipseHeight	Returns the length of the short axis of the ellipse of inertia.

EllipseWidth

Returns the length of the long axis of the ellipse of inertia.

FeretBox22Box

Returns the Feret box at orientation 22.5°.

FeretBox22Center

Returns the coordinates of the center of the Feret box oriented at 22.5°.

FeretBox22CenterX

Returns the abscissa of the center of the Feret box oriented at 22.5°.

FeretBox22CenterY

Returns the ordinate of the center of the Feret box oriented at 22.5°.

FeretBox22Height

Returns the height of the Feret box oriented at 22.5° (Feret diameter at 112.5°).

FeretBox22Width

Returns the width of the Feret box oriented at 22.5° (Feret diameter at 22.5°).

FeretBox45Box

Returns the Feret box at orientation 45°.

FeretBox45Center

Returns the coordinates of the center of the Feret box oriented at 45°.

FeretBox45CenterX

Returns the abscissa of the center of the Feret box oriented at 45°.

FeretBox45CenterY	Returns the ordinate of the center of the Feret box oriented at 45°.
FeretBox45Height	Returns the height of the Feret box oriented at 45° (Feret diameter at 135°).
FeretBox45Width	Returns the width of the Feret box oriented at 45° (Feret diameter at 45°).
FeretBox68Box	Returns the Feret box at orientation 67.5°.
FeretBox68Center	Returns the coordinates of the center of the Feret box oriented at 67.5°.
FeretBox68CenterX	Returns the abscissa of the center of the Feret box oriented at 67.5°.
FeretBox68CenterY	Returns the ordinate of the center of the Feret box oriented at 67.5°.
FeretBox68Height	Returns the height of the Feret box oriented at 67.5° (Feret diameter at 157.5°).
FeretBox68Width	Returns the width of the Feret box oriented at 67.5° (Feret diameter at 67.5°).
GravityCenter	Returns the gravity center of the coded element.

GravityCenterX	Returns the abscissa of the gravity center of the coded element.
GravityCenterY	Returns the ordinate of the gravity center of the coded element.
IsCodedElement	Tests whether the coded element is an object, a hole or a coded element.
IsHole	Tests whether the coded element is an object, a hole or a coded element.
IsObject	Tests whether the coded element is an object, a hole or a coded element.
LargestRun	Returns the length of the largest run inside the coded element.
LayerIndex	Returns the index of the layer in the coded image to which the coded element belongs.
LeftLimit	Returns the lowest (integer) X-coordinate of all the pixels of the coded element.
MinimumEnclosingRectangle	Returns the Minimum-Area Enclosing Rectangle.
MinimumEnclosingRectangleAngle	Returns the angle of the Minimum-Area Enclosing Rectangle.

MinimumEnclosingRectangleCenter

Returns the coordinates of the center of the Minimum-Area Enclosing Rectangle.

MinimumEnclosingRectangleCenterX

Returns the abscissa of the center of the Minimum-Area Enclosing Rectangle.

MinimumEnclosingRectangleCenterY

Returns the ordinate of the center of the Minimum-Area Enclosing Rectangle.

MinimumEnclosingRectangleHeight

Returns the height of the Minimum-Area Enclosing Rectangle.

MinimumEnclosingRectangleWidth

Returns the width of the Minimum-Area Enclosing Rectangle.

RightLimit

Returns the highest (integer) X-coordinate of all the pixels of the coded element.

RunCount

Returns the number of runs inside the coded element.

RunsIterator

Returns an iterator to the runs of the coded element.

SigmaX

Returns the centered moment of inertia around X (average squared X-deviation).

SigmaXX

Returns the centered cross moment of inertia (average X-deviation * Y-deviation).

SigmaXY

Returns the reduced, centered moment of inertia (around the principal inertia axis).

SigmaY

Returns the centered moment of inertia around Y (average squared Y-deviation).

SigmaYY

Returns the reduced, centered moment of inertia (around the secondary inertia axis).

TopLimit

M Returns the lowest (integer) Y-coordinate of all the pixels of the coded element.

e

thods

AsHole

Down-casts the coded element as a hole.

AsObject

Down-casts the coded element as an object.

ComputeConvexHull

Computes the convex hull of the coded element.

ComputeFeretBox	Computes the Feret box at a specific orientation.
ComputePixelGrayAverage	Computes the average gray-level value of the pixels of a given image over the coded element.
ComputePixelGrayDeviation	Computes the standard deviation of the gray-level values of a given image over the coded element.
ComputePixelGrayVariance	Computes the variance of the gray-level values of a given image over the coded element.
ComputePixelMax	Computes the maximum gray level of the pixels of a given image over the coded element.
ComputePixelMin	Computes the minimum gray level of the pixels of a given image over the coded element.
ComputeWeightedGravityCenter	Computes the gravity center of a given image over the coded element.
GetCentralMoment	Computes the central, two-dimensional moment of order (p,q) .
GetMoment	Computes the raw, two-dimensional moment of order (p,q) .
GetNormalizedCentralMoment	Computes the scale-invariant, central, two-dimensional moment of order (p,q) .

RenderMask

Creates a Flexible Mask from the coded element.

C

odedElement.Area

Returns the number of pixels inside the coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint Area  
{ get; }
```

Remarks

Equivalently, the area corresponds to the sum of the length of the runs of the coded element.

ECodedElement.AsHole

Down-casts the coded element as a hole.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EHole AsHole(  
    )
```

Remarks

This method throws an exception if the coded element is in fact an object.

ECodedElement.AsObject

Down-casts the coded element as an object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EObject AsObject(
)
```

Remarks

This method throws an exception if the coded element is in fact a hole.

ECodedElement.BottomLimit

Returns the highest (integer) Y-coordinate of all the pixels of the coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int BottomLimit
{ get; }
```

Remarks

For a coded element E, this value is defined as: $\left\lceil \max \left\{ y \mid (\exists x) (x, y) \in E \right\} \right\rceil$

ECodedElement.BoundingBox

Returns the bounding box of the coded element (Feret box at orientation 0°).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.ERotatedBoundingBox BoundingBox  
    { get; }
```

ECodedElement.BoundingBoxCenter

Returns the coordinates of the center of the bounding box of the coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint BoundingBoxCenter  
    { get; }
```

ECodedElement.BoundingBoxCenterX

Returns the abscissa of the center of the bounding box of the coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float BoundingBoxCenterX  
  
    { get; }
```

ECodedElement.BoundingBoxCenterY

Returns the ordinate of the center of the bounding box of the coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float BoundingBoxCenterY  
  
    { get; }
```

ECodedElement.BoundingBoxHeight

Returns the height of the bounding box (Feret diameter at 90°).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float BoundingBoxHeight  
  
    { get; }
```

ECodedElement.BoundingBoxWidth

Returns the width of the bounding box (Feret diameter at 0°).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float BoundingBoxWidth  
    { get; }
```

ECodedElement.ComputeConvexHull

Computes the convex hull of the coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void ComputeConvexHull(  
    Euresys.Open_eVision_2_10.EPathVector result  
)
```

Parameters

result

The output vector where to store the convex hull.

ECodedElement.ComputeFeretBox

Computes the Feret box at a specific orientation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.ERotatedBoundingBox ComputeFeretBox(  
    float angle  
)
```

Parameters

angle

The orientation of interest (in the current angle units).

ECodedElement.ComputePixelGrayAverage

Computes the average gray-level value of the pixels of a given image over the coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ComputePixelGrayAverage(  
    Euresys.Open_eVision_2_10.EROIBW8 image  
)
```

Parameters

image

The input image.

ECodedElement.ComputePixelGrayDeviation

Computes the standard deviation of the gray-level values of a given image over the coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float ComputePixelGrayDeviation(
    Euresys.Open_eVision_2_10.EROIBW8 image
)
```

Parameters

image

The input image.

ECodedElement.ComputePixelGrayVariance

Computes the variance of the gray-level values of a given image over the coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
double ComputePixelGrayVariance(
    Euresys.Open_eVision_2_10.EROIBW8 image
)
```

Parameters

image

The input image.

ECodedElement.ComputePixelMax

Computes the maximum gray level of the pixels of a given image over the coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EBW8 ComputePixelMax(
    Euresys.Open_eVision_2_10.EROIBW8 image
)
```

Parameters

image

The input image.

ECodedElement.ComputePixelMin

Computes the minimum gray level of the pixels of a given image over the coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EBW8 ComputePixelMin(
    Euresys.Open_eVision_2_10.EROIBW8 image
)
```

Parameters

image

The input image.

ECodedElement.ComputeWeightedGravityCenter

Computes the gravity center of a given image over the coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint ComputeWeightedGravityCenter (
    Euresys.Open_eVision_2_10.EROIBW8 image
)
```

Parameters

image

The input image.

ECodedElement.Contour

Returns the coordinates of the starting point of the countour of the coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint Contour
{ get; }
```

Remarks

More precisely, the leftmost pixel over the topmost row of the coded element is taken into consideration.

ECodedElement.ContourX

Returns the abscissa of the starting point of the countour of the coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int ContourX
{ get; }
```

ECodedElement.ContourY

Returns the ordinate of the starting point of the contour of the coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int ContourY
{ get; }
```

ECodedElement.Eccentricity

Returns the eccentricity of the ellipse of inertia.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float Eccentricity
{ get; }
```

Remarks

The ellipse of inertia is defined as the ellipse that has the same second order moments as the original coded element. The eccentricity is zero for circular objects and one for a line-shaped objects.

ECodedElement.ElementIndex

Returns the index of the coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint ElementIndex  
    { get; }
```

Remarks

If the coded element is an object, its index is relative to the layer to which it belongs. If the coded element is a hole, its index is relative to its parent object.

ECodedElement.EllipseAngle

Returns the angle of the ellipse of inertia.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float EllipseAngle  
    { get; }
```

Remarks

The ellipse of inertia is defined as the ellipse that has the same second order moments as the original coded element.

ECodedElement.EllipseHeight

Returns the length of the short axis of the ellipse of inertia.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float EllipseHeight  
    { get; }
```

Remarks

The ellipse of inertia is defined as the ellipse that has the same second order moments as the original coded element.

ECodedElement.EllipseWidth

Returns the length of the long axis of the ellipse of inertia.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float EllipseWidth  
    { get; }
```

Remarks

The ellipse of inertia is defined as the ellipse that has the same second order moments as the original coded element.

ECodedElement.FeretBox22Box

Returns the Feret box at orientation 22.5°.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.ERotatedBoundingBox FeretBox22Box  
{ get; }
```

ECodedElement.FeretBox22Center

Returns the coordinates of the center of the Feret box oriented at 22.5°.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint FeretBox22Center  
{ get; }
```

ECodedElement.FeretBox22CenterX

Returns the abscissa of the center of the Feret box oriented at 22.5°.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
```

```
float FeretBox22CenterX  
  
{ get; }
```

ECodedElement.FeretBox22CenterY

Returns the ordinate of the center of the Feret box oriented at 22.5°.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float FeretBox22CenterY  
  
{ get; }
```

ECodedElement.FeretBox22Height

Returns the height of the Feret box oriented at 22.5° (Feret diameter at 112.5°).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float FeretBox22Height  
  
{ get; }
```

ECodedElement.FeretBox22Width

Returns the width of the Feret box oriented at 22.5° (Feret diameter at 22.5°).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float FeretBox22Width  
    { get; }
```

ECodedElement.FeretBox45Box

Returns the Feret box at orientation 45°.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.ERotatedBoundingBox FeretBox45Box  
    { get; }
```

ECodedElement.FeretBox45Center

Returns the coordinates of the center of the Feret box oriented at 45°.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EPoint FeretBox45Center  
    { get; }
```

ECodedElement.FeretBox45CenterX

Returns the abscissa of the center of the Feret box oriented at 45°.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float FeretBox45CenterX  
    { get; }
```

ECodedElement.FeretBox45CenterY

Returns the ordinate of the center of the Feret box oriented at 45°.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float FeretBox45CenterY  
    { get; }
```

ECodedElement.FeretBox45Height

Returns the height of the Feret box oriented at 45° (Feret diameter at 135°).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float FeretBox45Height  
    { get; }
```

ECodedElement.FeretBox45Width

Returns the width of the Feret box oriented at 45° (Feret diameter at 45°).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float FeretBox45Width  
    { get; }
```

ECodedElement.FeretBox68Box

Returns the Feret box at orientation 67.5°.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.ERotatedBoundingBox FeretBox68Box  
  
{ get; }
```

ECodedElement.FeretBox68Center

Returns the coordinates of the center of the Feret box oriented at 67.5°.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EPoint FeretBox68Center  
  
{ get; }
```

ECodedElement.FeretBox68CenterX

Returns the abscissa of the center of the Feret box oriented at 67.5°.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float FeretBox68CenterX  
  
{ get; }
```

ECodedElement.FeretBox68CenterY

Returns the ordinate of the center of the Feret box oriented at 67.5°.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float FeretBox68CenterY  
    { get; }
```

ECodedElement.FeretBox68Height

Returns the height of the Feret box oriented at 67.5° (Feret diameter at 157.5°).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float FeretBox68Height  
    { get; }
```

ECodedElement.FeretBox68Width

Returns the width of the Feret box oriented at 67.5° (Feret diameter at 67.5°).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float FeretBox68Width
{ get; }
```

ECodedElement.GetCentralMoment

Computes the central, two-dimensional moment of order (p,q).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float GetCentralMoment(
    uint p,
    uint q
)
```

Parameters

- p
Order of the moment along the X-axis.
- q
Order of the moment along the Y-axis.

Remarks

$$\mu_{pq} = \sum_x \sum_y (x - \bar{x})^p (y - \bar{y})^q I(x, y)$$

$$\mu_{pq} = \sum_x \sum_y (x - \bar{x})^p (y - \bar{y})^q I(x, y)$$

ECodedElement.GetMoment

Computes the raw, two-dimensional moment of order (p,q).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
double GetMoment(  
    uint p,  
    uint q  
)
```

Parameters

p
Order of the moment along the X-axis.

q
Order of the moment along the Y-axis.

Remarks

$$M_{pq} = \sum_x \sum_y x^p y^q I(x, y)$$

<latex>M_{pq} = \sum_x \sum_y x^p y^q I(x,y)</latex>

ECodedElement.GetNormalizedCentralMoment

Computes the scale-invariant, central, two-dimensional moment of order (p,q).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
float GetNormalizedCentralMoment(  
    uint p,  
    uint q  
)
```

Parameters

p
Order of the moment along the X-axis.

q

Order of the moment along the Y-axis.

Remarks

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00} \left(1 + \frac{p+q}{2}\right)}$$

ECodedElement.GravityCenter

Returns the gravity center of the coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint GravityCenter  
  
{ get; }
```

ECodedElement.GravityCenterX

Returns the abscissa of the gravity center of the coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float GravityCenterX  
  
{ get; }
```

Remarks

$$\frac{\sum_{(x,y) \in E} x}{\sum_{(x,y) \in E} 1}$$

For a coded element E, this value is defined as: $\frac{\sum_{(x,y) \in E} x}{\sum_{(x,y) \in E} 1}$

ECodedElement.GravityCenterY

Returns the ordinate of the gravity center of the coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float GravityCenterY  
{ get; }
```

Remarks

$$\frac{\sum_{(x,y) \in E} y}{\sum_{(x,y) \in E} 1}$$

For a coded element E, this value is defined as: $\frac{\sum_{(x,y) \in E} y}{\sum_{(x,y) \in E} 1}$

ECodedElement.IsCodedElement

Tests whether the coded element is an object, a hole or a coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
bool IsCodedElement
```

```
{ get; }
```

ECodedElement.IsHole

Tests whether the coded element is an object, a hole or a coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
bool IsHole
```

```
{ get; }
```

ECodedElement.IsObject

Tests whether the coded element is an object, a hole or a coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
bool IsObject
```

```
{ get; }
```

ECodedElement.LargestRun

Returns the length of the largest run inside the coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint LargestRun  
    { get; }
```

ECodedElement.LayerIndex

Returns the index of the layer in the coded image to which the coded element belongs.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint LayerIndex  
    { get; }
```

Remarks

If the coded element is a hole, its layer index is defined as that of its parent object.

ECodedElement.LeftLimit

Returns the lowest (integer) X-coordinate of all the pixels of the coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int LeftLimit
```

```
{ get; }
```

Remarks

For a coded element E, this value is defined as: $\left\lfloor \min \left\{ x \mid (\exists y) (x, y) \in E \right\} \right\rfloor$

ECodedElement.MinimumEnclosingRectangle

Returns the Minimum-Area Enclosing Rectangle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.ERotatedBoundingBox MinimumEnclosingRectangle  
{ get; }
```

Remarks

The Minimum-Area Enclosing Rectangle is defined as the Feret box with the minimum surface among all the possible orientations.

ECodedElement.MinimumEnclosingRectangleAngle

Returns the angle of the Minimum-Area Enclosing Rectangle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float MinimumEnclosingRectangleAngle  
{ get; }
```

Remarks

The angle always lies in the range $[0 ; \pi[$.

ECodedElement.MinimumEnclosingRectangleCenter

Returns the coordinates of the center of the Minimum-Area Enclosing Rectangle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint MinimumEnclosingRectangleCenter  
{ get; }
```

ECodedElement.MinimumEnclosingRectangleCenterX

Returns the abscissa of the center of the Minimum-Area Enclosing Rectangle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float MinimumEnclosingRectangleCenterX  
{ get; }
```

ECodedElement.MinimumEnclosingRectangleCenterY

Returns the ordinate of the center of the Minimum-Area Enclosing Rectangle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float MinimumEnclosingRectangleCenterY  
    { get; }
```

ECodedElement.MinimumEnclosingRectangleHeight

Returns the height of the Minimum-Area Enclosing Rectangle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float MinimumEnclosingRectangleHeight  
    { get; }
```

ECodedElement.MinimumEnclosingRectangleWidth

Returns the width of the Minimum-Area Enclosing Rectangle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float MinimumEnclosingRectangleWidth
{ get; }
```

ECodedElement.RenderMask

Creates a Flexible Mask from the coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void RenderMask(
    Euresys.Open_eVision_2_10.EROIBW8 destination,
    int offsetX,
    int offsetY
)
void RenderMask(
    Euresys.Open_eVision_2_10.EROIBW8 destination
)
```

Parameters

destination

The image in which the generated mask will be stored.

offsetX

The X-offset that must be applied to bring the zero X-coordinate in the coded image on the first column of the result image (defaults to zero).

offsetY

The Y-offset that must be applied to bring the zero Y-coordinate in the coded image on the first row of the result image (defaults to zero).

Remarks

The size of the result image will not be changed: It must be properly sized beforehand.

ECodedElement.RightLimit

Returns the highest (integer) X-coordinate of all the pixels of the coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int RightLimit  
{ get; }
```

Remarks

For a coded element E, this value is defined as: $\left\lceil \max \left\{ x \mid (\exists y) (x, y) \in E \right\} \right\rceil$

ECodedElement.RunCount

Returns the number of runs inside the coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint RunCount  
{ get; }
```

ECodedElement.RunsIterator

Returns an iterator to the runs of the coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EObjectRunsIterator RunsIterator  
    { get; }
```

ECodedElement.SigmaX

Returns the centered moment of inertia around X (average squared X-deviation).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float SigmaX  
    { get; }
```

ECodedElement.SigmaXX

Returns the centered cross moment of inertia (average X-deviation * Y-deviation).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float SigmaXX  
    { get; }
```

ECodedElement.SigmaXY

Returns the reduced, centered moment of inertia (around the principal inertia axis).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float SigmaXY  
{ get; }
```

ECodedElement.SigmaY

Returns the centered moment of inertia around Y (average squared Y-deviation).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float SigmaY  
{ get; }
```

ECodedElement.SigmaYY

Returns the reduced, centered moment of inertia (around the secondary inertia axis).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float SigmaYY
{ get; }
```

ECodedElement.TopLimit

Returns the lowest (integer) Y-coordinate of all the pixels of the coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int TopLimit
{ get; }
```

Remarks

For a coded element E, this value is defined as: $\lfloor \min \{ y \mid (\exists x) (x, y) \in E \} \rfloor$

4.47. ECodedImage Class

This class handles runs, objects and features in EasyObject.

Remarks

These entities are stored into three separate dynamic lists for efficient storage. This class pertains to the EasyObject legacy API and should not be used for new developments. It has been replaced by [ECodedImage2](#).

Namespace: Euresys.Open_eVision_2_10

Properties

BlackClass

Black class index (below the lower threshold).

Connexity

Connexity mode, that is how neighboring pixels are considered to belong to the same objects.

Continuous

Flag indicating whether the objects have to be built in the continuous mode or in the normal mode.

CurrentObjPtr

Pointer to the current objects list item, or **NULL**.

CurrentRunPtr

Pointer to the current run list item, or **NULL**.

DrawDiagonals

Flag indicating whether the limit rectangle diagonals must be drawn or not.

FirstObjPtr

Pointer to the first objects list item, or **NULL**.

HighColorThreshold

Upper threshold (color) used for image segmentation.

HighImage

Image used as an adaptive upper threshold.

HighThreshold

Upper threshold (gray level) used for image segmentation.

LastObjPtr	Pointer to the last objects list item, or NULL .
LimitAngle	Angle of the skewed bounding box feature, in the current angle unit.
LowColorThreshold	Lower threshold (color) used for image segmentation.
LowImage	Image used as an adaptive lower threshold.
LowThreshold	Lower threshold (gray level) used for image segmentation.
MaxObjects	Maximum number of objects to look for.
NeutralClass	Neutral class index (between both thresholds).
NumFeatures	Number of features currently in use.
NumHoleRuns	Total number of hole runs in the list of object runs.
NumObjects	Number of objects in the coded image.
NumRuns	Total number of runs in the list of object runs.
NumSelectedObjects	Number of objects currently selected.

Threshold

Threshold mode (gray level) used for image segmentation.

ThresholdImage

Single threshold used for image segmentation.

TrueThreshold

Absolute threshold level, when using a single threshold.

WhiteClass

M White class index (above the upper threshold).

e

thods

AddFeat

Adds a feature to the list of features.

AnalyseObjects

After an image segmentation (see [ECodedImage::BuildObjects](#)), computes the values of given "features", i.e. geometric parameters.

BlankFeatures

Resets all values of all features.

BuildHoles

Creates holes.

BuildLabeledObjects

Segments an image into connected blobs comprised of pixels of the same class.

BuildLabeledRuns

Extracts the runs from the image by comparing adjacent pixel values.

BuildObjects

Groups runs to form separated objects (connected blobs), from runs detected by [ECodedImage::BuildRuns](#) or from a source image/ROI.

BuildRuns

Converts the specified ROI to classes, and extracts the runs from it.

DrawObject

Draws the designated object in solid color.

DrawObjectFeature

Draws a graphical representation of a feature of the designated object in solid color.

DrawObjectFeatureWithCurrentPen

Draws a graphical representation of a feature of the designated object in solid color.

DrawObjects

Draws all objects in solid color.

DrawObjectsFeature

Draws a graphical representation of a feature.

DrawObjectsFeatureWithCurrentPen

Draws a graphical representation of a feature.

DrawObjectsWithCurrentPen

Draws all objects in solid color.

DrawObjectWithCurrentPen

Draws the designated object in solid color.

ECodedImage

Constructs a void coded image.

FeatureAverage

Computes the average of the features of all currently selected objects.

FeatureDeviation

Computes the average and standard deviation of the features of all currently selected objects.

FeatureMaximum

Computes the maximum of the features of all currently selected objects.

FeatureMinimum

Computes the minimum of the features of all currently selected objects.

FeatureVariance

Computes the average and variance of the features of all currently selected objects.

GetCurrentObjData

Returns the data of the current object.

GetCurrentRunData

Returns the data of the current run.

GetFeatData

Gets the [EFeatureData](#) associated to a given feature list item.

GetFeatDataSize

Returns the data size of the specified feature.

GetFeatDataType

Returns the data type of the specified feature.

GetFeatNum

Returns the code number of the specified feature.

GetFeatPtrByNum

Returns a pointer to the feature list item for a given feature number, or **NULL**.

GetFeatSize

Returns the size (number of elements) of the feature array for the specified feature.

GetFirstHole

Returns a pointer to the first hole related to the specified object.

GetFirstObjData

Moves the cursor to the first object and returns the associated data.

GetFirstRunData

Moves the cursor to the first run and returns the associated data.

GetFirstRunPtr

Pointer to the first run list item, or **NULL**.

GetHoleParentObject

Returns a pointer to the real object including the specified hole.

GetLastObjData

Moves the cursor to the last object and returns the associated data.

GetLastRunData

Moves the cursor to the last run and returns the associated data.

GetLastRunPtr

Pointer to the last run list item, or **NULL**.

[GetNextHole](#)

Returns a pointer to the hole following the specified hole, in the objects list.

[GetNextObjData](#)

Moves the cursor to the next object and returns the associated data.

[GetNextObjPtr](#)

Returns a pointer to the next objects list item, or **NULL**.

[GetNextRunData](#)

Moves the cursor to the next run and returns the associated data.

[GetNextRunPtr](#)

Returns a pointer to the next run list item, or **NULL**.

[GetNumHoles](#)

Returns the number of holes related to the specified object.

[GetNumObjectRuns](#)

Returns the number of runs comprised in a given object.

[GetObjDataPtr](#)

Gets the [EObjectData](#) associated to a given objects list item.

[GetObjectData](#)

If an object identification number is given, moves the cursor to the object with a given identification number and returns the associated data. If a list item is given, returns the object data associated with an object list item (cf. [EListItem](#)). See also [ECodedImage::GetCurrentObjData](#).

[GetObjectFeature](#)

Allows retrieving the value of a feature of a given object.

GetObjFirstRunPtr

Returns a pointer to the first run list item of an object.

GetObjLastRunPtr

Returns a pointer to the last run list item of an object.

GetObjPtr

Returns a pointer to the given objects list item.

GetObjPtrByCoordinates

Returns a pointer to the objects list item that contains the point of given coordinates, or **NULL**.

GetObjPtrByPos

Returns a pointer to the objects list item of given absolute position, or **NULL**.

GetPreviousObjData

Moves the cursor to the previous object and returns the associated data.

GetPreviousObjPtr

Returns a pointer to the previous objects list item, or **NULL**.

GetPreviousRunData

Moves the cursor to the previous run and returns the associated data.

GetPreviousRunPtr

Returns a pointer to the previous run list item, or **NULL**.

GetRunData

Returns the run data associated to the specified run.

GetRunDataPtr

Gets the [ERunData](#) associated to a given run list item.

GetRunPtr

Returns a pointer to the run list item of given absolute position, or **NULL**.

GetRunPtrByCoordinates

Returns a pointer to the run list item that contains the point of given coordinates, or **NULL**.

IsHole

Returns **TRUE** if the specified object is a hole, **FALSE** otherwise.

IsObjectSelected

Sets **bSelected** to **TRUE** if an object is selected.

ObjectConvexHull

Computes the convex hull of an object and stores it in a **EPathVector**.

RemoveAllFeats

Deletes all features from the features list.

RemoveAllObjects

Deletes all objects from the objects list.

RemoveAllRuns

Deletes all runs from the runs list.

RemoveHoles

Permanently erases, from the objects list, holes related to the specified object.

RemoveObject

Deletes an object from the objects list.

RemoveRun

Deletes a run from the runs list.

ResetContinuousMode

When the continuous mode is activated, this method resets the sequence of images.

SelectAllObjects

Selects all objects.

SelectHoles

Selects the holes related to the specified object.

SelectObject

Selects an object.

SelectObjectsUsingFeature

Selects or deselects objects, according to the value of a specified feature.

SelectObjectsUsingPosition

Selects or deselects objects, using a delimiting rectangle.

SetFeatInfo

Sets the appropriate data size and type for a predefined feature.

SetFirstRunPtr

Sets the first run list item of an object.

SetLastRunPtr

Sets the last run list item of an object.

SortObjectsUsingFeature

Sorts objects according to the value of some feature.

UnselectAllObjects

Deselects all objects.

[UnselectHoles](#)

Unselects the holes related to the specified object.

[UnselectObject](#)

Deselects an object.

C

EodedImage.AddFeat

Adds a feature to the list of features.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddFeat(
    ref Euresys.Open_eVision_2_10.EFeatureData feature,
    int numberOfObjects
)
```

Parameters

feature

Pointer to an [EFeatureData](#) describing the feature.

numberOfObjects

Number of objects for which the feature will be stored.

ECodedImage.AnalyseObjects

After an image segmentation (see [ECodedImage::BuildObjects](#)), computes the values of given "features", i.e. geometric parameters.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AnalyseObjects (
    Euresys.Open_eVision_2_10.ELegacyFeature feature1,
    Euresys.Open_eVision_2_10.ELegacyFeature feature2,
    Euresys.Open_eVision_2_10.ELegacyFeature feature3,
    Euresys.Open_eVision_2_10.ELegacyFeature feature4,
    Euresys.Open_eVision_2_10.ELegacyFeature feature5,
    Euresys.Open_eVision_2_10.ELegacyFeature feature6,
    Euresys.Open_eVision_2_10.ELegacyFeature feature7,
    Euresys.Open_eVision_2_10.ELegacyFeature feature8,
    Euresys.Open_eVision_2_10.ELegacyFeature feature9,
    Euresys.Open_eVision_2_10.ELegacyFeature feature10
)
```

Parameters

feature1

Feature code, as defined by [ELegacyFeature](#).

feature2

Feature code, as defined by [ELegacyFeature](#).

feature3

Feature code, as defined by [ELegacyFeature](#).

feature4

Feature code, as defined by [ELegacyFeature](#).

feature5

Feature code, as defined by [ELegacyFeature](#).

feature6

Feature code, as defined by [ELegacyFeature](#).

feature7

Feature code, as defined by [ELegacyFeature](#).

feature8

Feature code, as defined by [ELegacyFeature](#).

feature9

Feature code, as defined by [ELegacyFeature](#).

feature10

Feature code, as defined by [ELegacyFeature](#).

ECodedImage.BlackClass

Black class index (below the lower threshold).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
short BlackClass
{ get; set; }
```

Remarks

Non zero when the black runs (below the lower threshold) are coded. **0** means "do not code this class". <!-- **1** by default. -->

ECodedImage.BlankFeatures

Resets all values of all features.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void BlankFeatures (
)
```

ECodedImage.BuildHoles

Creates holes.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void BuildHoles (
)
void BuildHoles (
    Euresys.Open_eVision_2_10.EListItem object_
)
```

Parameters

object_

Pointer to the objects list item, for which the holes have to be computed.

Remarks

If no argument, the holes are related to all the previously selected real objects. If holes already exist (resulting from a previous call to the [ECodedImage::BuildHoles](#) function), they will be removed from the objects list before the new hole building. Otherwise, the holes are related only to the specified object. Previously created holes are not removed before the new holes are built. If holes related to **object** have already been constructed, they won't be recreated. If **object** is a hole or is **NULL**, no hole will be built. The newly created holes will be added to the list of the objects found in the image. Building holes requires two preliminary steps: the construction of real objects and the selection of objects on which the hole detection has to be performed. At the end of the object construction, all the objects are selected.

ECodedImage.BuildLabeledObjects

Segments an image into connected blobs comprised of pixels of the same class.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void BuildLabeledObjects (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage
)
```

```
void BuildLabeledObjects (  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage  
)
```

Parameters

sourceImage

Pointer to a source ROI.

Remarks

Uses [EBW8 \(EBW16\)](#) information for class indices, i.e. 255 (65,535) possible classes. Class 0 is not coded. Building objects is the process of grouping pixels from an image to form connected blobs. The pixels are assigned class indices based either on thresholding ([ECodedImage::BuildObjects](#)) or on the pixel values themselves ([BuildLabeledObjects](#)). A blob is a set of connected pixels of the same class.

ECodedImage.BuildLabeledRuns

Extracts the runs from the image by comparing adjacent pixel values.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void BuildLabeledRuns (  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage  
)  
  
void BuildLabeledRuns (  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

Remarks

Uses [EBW8 \(EBW16\)](#) information for class indices, i.e. 255 (65,535) possible classes. Class 0 is not coded. Building runs is the process of grouping pixels from an image to form horizontal segments. The pixels are assigned class indices based either on thresholding ([ECodedImage::BuildRuns](#)) or on the pixel values themselves (**BuildLabeledRuns**). A run is a set of horizontally connected pixels of the same class.

ECodedImage.BuildObjects

Groups runs to form separated objects (connected blobs), from runs detected by [ECodedImage::BuildRuns](#) or from a source image/ROI.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void BuildObjects (
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage
)
void BuildObjects (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage
)
void BuildObjects (
    Euresys.Open_eVision_2_10.EROIC24 sourceImage
)
void BuildObjects (
)
```

Parameters

sourceImage

Pointer to a source ROI.

Remarks

Without argument, the method groups the runs detected by [ECodedImage::BuildRuns](#) to form separate objects, i.e. connected components. With a source ROI as argument, the method segments it into connected blobs comprised of pixels of the same class. The [EROIBW8](#) parameter is converted to white/neutral/black classes, using two thresholds. The [EROIC24](#) parameter is converted to white/black classes, using two color thresholds. Then, the method extracts runs from them, and groups the runs to form separate objects, i.e. connected components. Building objects is the process of grouping pixels from an image to form connected blobs. The pixels are assigned class indices based either on thresholding (**BuildObjects**) or on the pixel values themselves ([ECodedImage::BuildLabeledObjects](#)). A blob is a set of connected pixels of the same class.

ECodedImage.BuildRuns

Converts the specified ROI to classes, and extracts the runs from it.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void BuildRuns (
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage
)
void BuildRuns (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage
)
void BuildRuns (
    Euresys.Open_eVision_2_10.EROIC24 sourceImage
)
```

Parameters

sourceImage

Pointer to the source ROI.

Remarks

The [EROIBW8](#) parameter is converted to white/neutral/black classes, using two thresholds. The [EROIC24](#) parameter is converted to white/black classes, using two color thresholds. Then, the method extracts runs from them. Building runs is the process of grouping pixels from an image to form horizontal segments. The pixels are assigned class indices based either on thresholding (**BuildRuns**) or on the pixel values themselves ([ECodedImage::BuildLabeledRuns](#)). A run is a set of horizontally connected pixels of the same class.

ECodedImage.Connexity

Connexity mode, that is how neighboring pixels are considered to belong to the same objects.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EConnexity Connexity  
    { get; set; }
```

ECodedImage.Continuous

Flag indicating whether the objects have to be built in the continuous mode or in the normal mode.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool Continuous  
    { get; set; }
```

Remarks

TRUE if objects are built in the continuous mode, **FALSE** if objects are built in the normal mode.

ECodedImage.CurrentObjPtr

Pointer to the current objects list item, or **NULL**.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EListItem CurrentObjPtr
{ get; }
```

ECodedImage.CurrentRunPtr

Pointer to the current run list item, or **NULL**.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EListItem CurrentRunPtr
{ get; }
```

ECodedImage.DrawDiagonals

Flag indicating whether the limit rectangle diagonals must be drawn or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool DrawDiagonals
{ get; set; }
```

Remarks

If **TRUE** (default), diagonals are drawn.

ECodedImage.DrawObject

Draws the designated object in solid color.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void DrawObject(
    IntPtr graphicContext,
    int objectNumber,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void DrawObject(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EListItem object_,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void DrawObject(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    int objectNumber,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```



```

void DrawObject(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.EListItem object_,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void DrawObject(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    int objectNumber,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void DrawObject(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.EListItem object_,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

```

Parameters

graphicContext

Handle of the device context on which to draw.

objectNumber

Number of the object to be drawn.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0**, the default value, the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

object_

Pointer to the list item (from the objects list) corresponding to the object to be drawn.

color

The color in which to draw the overlay.

Remarks

Drawing is done in the device context associated to the desired window. The current pen is used. The objects can be specified either by number of by list pointer.

ECodedImage.DrawObjectFeature

Draws a graphical representation of a feature of the designated object in solid color.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawObjectFeature(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ELegacyFeature feature,
    int objectNumber,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void DrawObjectFeature(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ELegacyFeature feature,
    Euresys.Open_eVision_2_10.EListItem object_,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

```

void DrawObjectFeature(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.ELegacyFeature feature,
    Euresys.Open_eVision_2_10.EListItem object_,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void DrawObjectFeature(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.ELegacyFeature feature,
    int objectNumber,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void DrawObjectFeature(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.ELegacyFeature feature,
    int objectNumber,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void DrawObjectFeature(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.ELegacyFeature feature,
    Euresys.Open_eVision_2_10.EListItem object_,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

```

Parameters

graphicContext

Handle of the device context on which to draw.

feature

Feature to be drawn, as defined by [ELegacyFeature](#) (use any value matching the aforementioned features, f.i. [EllipseWidth](#) will draw the ellipse of inertia).

objectNumber

Number of the object to be drawn.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0**, the default value, the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

object_

Pointer to the list item (from the objects list) corresponding to the object to be drawn.

color

The color in which to draw the overlay.

Remarks

Drawing is done in the device context associated to the desired window. The current pen is used. The objects can be specified either by number of by list pointer. If a required feature has not been computed for some object, nothing is drawn and no error message is issued!

ECodedImage.DrawObjectFeatureWithCurrentPen

Draws a graphical representation of a feature of the designated object in solid color.

Namespace: Euresys.Open_eVision_2_10

[C#]

```

void DrawObjectFeatureWithCurrentPen (
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ELegacyFeature feature,
    Euresys.Open_eVision_2_10.EListItem objectNumber,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void DrawObjectFeatureWithCurrentPen (
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ELegacyFeature feature,
    int objectNumber,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

```

Parameters

graphicContext

Handle of the device context on which to draw.

feature

Feature to be drawn, as defined by [ELegacyFeature](#) (use any value matching the aforementioned features, f.i. [EllipseWidth](#) will draw the ellipse of inertia).

objectNumber

Number of the object to be drawn.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0**, the default value, the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

Remarks

Drawing is done in the device context associated to the desired window. The current pen is used. The objects can be specified either by number of by list pointer. If a required feature has not been computed for some object, nothing is drawn and no error message is issued!

ECodedImage.DrawObjects

Draws all objects in solid color.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void DrawObjects(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ESelectionFlag selectionFlag,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void DrawObjects(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ERGBColor color,  
    Euresys.Open_eVision_2_10.ESelectionFlag selectionFlag,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void DrawObjects(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.ESelectionFlag selectionFlag,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

selectionFlag

Tells how the selected/unselected state of the objects is handled, as defined by [ESelectionFlag](#).

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

color

The color in which to draw the overlay.

Remarks

Drawing is done in the device context associated to the desired window. The current pen is used for all objects (to vary the colors, draw the objects separately using the [ECodedImage::DrawObject](#) method instead). Optionally, only the selected or deselected objects can be drawn.

ECodedImage.DrawObjectsFeature

Draws a graphical representation of a feature.

Namespace: Euresys.Open_eVision_2_10

[C#]

```

void DrawObjectsFeature(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ELegacyFeature feature,
    Euresys.Open_eVision_2_10.ESelectionFlag selectionFlag,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void DrawObjectsFeature(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.ELegacyFeature feature,
    Euresys.Open_eVision_2_10.ESelectionFlag selectionFlag,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void DrawObjectsFeature(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.ELegacyFeature feature,
    Euresys.Open_eVision_2_10.ESelectionFlag selectionFlag,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

```

Parameters

graphicContext

Handle of the device context on which to draw.

feature

Feature to be drawn, as defined by [ELegacyFeature](#) (use any value matching the aforementioned features, f.i. [EllipseWidth](#) will draw the ellipse of inertia).

selectionFlag

Tells how the selected/unselected state of the objects is handled, as defined by [ESelectionFlag](#).

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

color

The color in which to draw the overlay.

Remarks

Drawing is done in the device context associated to the desired window. The current pen is used for all objects (to vary the colors, draw the objects separately using the [ECodedImage::DrawObjectFeature](#) method instead). Optionally, only the selected or deselected objects can be drawn. Only the following features can be drawn: * [GravityCenter](#): upright cross; * [Centroid](#): skewed cross; * [Limit](#): upright bounding rectangle with diagonals; * [Limit45](#): skewed bounding rectangle with diagonals; * [EllipseWidth](#): ellipse of inertia (edge and main axis). If a required feature has not been computed for some object, nothing is drawn and no error message is issued!

ECodedImage.DrawObjectsFeatureWithCurrentPen

Draws a graphical representation of a feature.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawObjectsFeatureWithCurrentPen(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ELegacyFeature feature,
    Euresys.Open_eVision_2_10.ESelectionFlag selectionFlag,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

feature

Feature to be drawn, as defined by [ELegacyFeature](#) (use any value matching the aforementioned features, f.i. [EllipseWidth](#) will draw the ellipse of inertia).

selectionFlag

Tells how the selected/unselected state of the objects is handled, as defined by [ESelectionFlag](#).

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

Remarks

Drawing is done in the device context associated to the desired window. The current pen is used for all objects (to vary the colors, draw the objects separately using the [ECodedImage::DrawObjectFeature](#) method instead). Optionally, only the selected or deselected objects can be drawn. Only the following features can be drawn: * [GravityCenter](#): upright cross; * [Centroid](#): skewed cross; * [Limit](#): upright bounding rectangle with diagonals; * [Limit45](#): skewed bounding rectangle with diagonals; * [EllipseWidth](#): ellipse of inertia (edge and main axis). If a required feature has not been computed for some object, nothing is drawn and no error message is issued!

ECodedImage.DrawObjectsWithCurrentPen

Draws all objects in solid color.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawObjectsWithCurrentPen (
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ESelectionFlag selectionFlag,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

selectionFlag

Tells how the selected/unselected state of the objects is handled, as defined by [ESelectionFlag](#).

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

Remarks

Drawing is done in the device context associated to the desired window. The current pen is used for all objects (to vary the colors, draw the objects separately using the [ECodedImage::DrawObject](#) method instead). Optionally, only the selected or deselected objects can be drawn.

ECodedImage.DrawObjectWithCurrentPen

Draws the designated object in solid color.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawObjectWithCurrentPen (
    IntPtr graphicContext,
    int objectNumber,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

```
void DrawObjectWithCurrentPen(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EListItem object_,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

objectNumber

Number of the object to be drawn.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0**, the default value, the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

object_

Pointer to the list item (from the objects list) corresponding to the object to be drawn.

Remarks

Drawing is done in the device context associated to the desired window. The current pen is used. The objects can be specified either by number of by list pointer.

ECodedImage.ECodedImage

Constructs a void coded image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ECodedImage (
    Euresys.Open_eVision_2_10.ECodedImage other
)

void ECodedImage (
)
```

Parameters

other

-

ECodedImage.FeatureAverage

Computes the average of the features of all currently selected objects.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void FeatureAverage (
    Euresys.Open_eVision_2_10.ELegacyFeature feature,
    out float average
)
```

Parameters

feature

Feature code, as defined by [ELegacyFeature](#).

average

Reference to the feature average.

Remarks

This measures the central tendency of a population of objects.

ECodedImage.FeatureDeviation

Computes the average and standard deviation of the features of all currently selected objects.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void FeatureDeviation(
    Euresys.Open_eVision_2_10.ELegacyFeature feature,
    out float average,
    out float deviation
)
```

Parameters

feature

Feature code, as defined by [ELegacyFeature](#).

average

Reference to the feature average.

deviation

Reference to the feature standard deviation.

ECodedImage.FeatureMaximum

Computes the maximum of the features of all currently selected objects.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void FeatureMaximum(  
    Euresys.Open_eVision_2_10.ELegacyFeature feature,  
    out float maximum  
)
```

Parameters

feature

Feature code, as defined by [ELegacyFeature](#).

maximum

Reference to the feature maximum.

ECodedImage.FeatureMinimum

Computes the minimum of the features of all currently selected objects.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void FeatureMinimum(  
    Euresys.Open_eVision_2_10.ELegacyFeature feature,  
    out float minimum  
)
```

Parameters

feature

Feature code, as defined by [ELegacyFeature](#).

minimum

Reference to the feature minimum.

ECodedImage.FeatureVariance

Computes the average and variance of the features of all currently selected objects.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void FeatureVariance(
    Euresys.Open_eVision_2_10.ELegacyFeature feature,
    out float average,
    out float variance
)
```

Parameters

feature

Feature code, as defined by [ELegacyFeature](#).

average

Reference to the feature average.

variance

Reference to the feature variance.

Remarks

This measures the central tendency and the dispersion of a population of objects.

ECodedImage.FirstObjPtr

Pointer to the first objects list item, or **NULL**.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```



```
Euresys.Open_eVision_2_10.EListItem FirstObjPtr  
{ get; }
```

ECodedImage.GetCurrentObjData

Returns the data of the current object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void GetCurrentObjData (  
    out Euresys.Open_eVision_2_10.EObjectData objectData  
)
```

Parameters

objectData

Pointer to an [EObjectData](#) structure to receive the data.

Remarks

ECodedImage.GetCurrentRunData

Returns the data of the current run.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void GetCurrentRunData (
    out Euresys.Open_eVision_2_10.ERunData run
)
```

Parameters

run

Pointer to an [ERunData](#) to receive the data.

ECodedImage.GetFeatData

Gets the [EFeatureData](#) associated to a given feature list item.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool GetFeatData (
    Euresys.Open_eVision_2_10.EListItem currentFeature,
    out Euresys.Open_eVision_2_10.EFeatureData featureData
)
```

Parameters

currentFeature

Pointer to the feature list item.

featureData

Pointer to a [EFeatureData](#) to receive the data.

ECodedImage.GetFeatDataSize

Returns the data size of the specified feature.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EDataSize GetFeatDataSize(
    int position
)
Euresys.Open_eVision_2_10.EDataSize GetFeatDataSize(
    Euresys.Open_eVision_2_10.EListItem currentFeature
)
```

Parameters

position

Absolute position in the features list, counting from **0** on.

currentFeature

Pointer to the feature list item.

Remarks

The features data sizes are defined in [EDataSize](#).

ECodedImage.GetFeatDataType

Returns the data type of the specified feature.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EDataType GetFeatDataType(
    int position
)
Euresys.Open_eVision_2_10.EDataType GetFeatDataType(
    Euresys.Open_eVision_2_10.EListItem currentFeature
)
```

Parameters

position

Absolute position in the features list, counting from **0** on.

currentFeature

Pointer to the feature list item.

Remarks

The features data types are defined in [EDataType](#).

ECodedImage.GetFeatNum

Returns the code number of the specified feature.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int GetFeatNum(
    int position
)

int GetFeatNum(
    Euresys.Open_eVision_2_10.EListItem currentFeature
)
```

Parameters

position

Absolute position in the features list, counting from **0** on.

currentFeature

Pointer to the feature list item.

Remarks

The features code numbers are defined in [ELegacyFeature](#).

ECodedImage.GetFeatPtrByNum

Returns a pointer to the feature list item for a given feature number, or **NULL**.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EListItem GetFeatPtrByNum(
    int numFeat
)
```

Parameters

numFeat

Feature number, as defined by [ELegacyFeature](#).

ECodedImage.GetFeatSize

Returns the size (number of elements) of the feature array for the specified feature.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int GetFeatSize(
    int position
)

int GetFeatSize(
    Euresys.Open_eVision_2_10.EListItem currentFeature
)
```

Parameters

position

Absolute position in the features list, counting from **0** on.

currentFeature

Pointer to the feature list item.

ECodedImage.GetFirstHole

Returns a pointer to the first hole related to the specified object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EListItem GetFirstHole(
    Euresys.Open_eVision_2_10.EListItem parentObject
)
```

Parameters

parentObject

Pointer to the objects list item, for which the first hole has to be pointed out.

Remarks

If **parentObject** refers to a hole (instead of a real object) or to an object comprising no hole, the [ECodedImage::GetFirstHole](#) function returns **NULL**.

ECodedImage.GetFirstObjData

Moves the cursor to the first object and returns the associated data.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetFirstObjData(
    out Euresys.Open_eVision_2_10.EObjectData object_
)
```

Parameters

object_

Pointer to an [EObjectData](#) to receive the data.

Remarks

ECodedImage.GetFirstRunData

Moves the cursor to the first run and returns the associated data.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetFirstRunData (
    out Euresys.Open_eVision_2_10.ERunData run
)
```

Parameters

run

Pointer to an [ERunData](#) to receive the data.

ECodedImage.GetFirstRunPtr

Pointer to the first run list item, or **NULL**.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EListItem GetFirstRunPtr (
)
```

ECodedImage.GetHoleParentObject

Returns a pointer to the real object including the specified hole.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EListItem GetHoleParentObject(
    Euresys.Open_eVision_2_10.EListItem hole
)
```

Parameters

hole

Pointer to the hole list item, for which the parent object has to be pointed out.

ECodedImage.GetLastObjData

Moves the cursor to the last object and returns the associated data.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetLastObjData (
    out Euresys.Open_eVision_2_10.EObjectData object_
)
```

Parameters

object_

Pointer to an [EObjectData](#) structure to receive the data.

ECodedImage.GetLastRunData

Moves the cursor to the last run and returns the associated data.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetLastRunData (
    out Euresys.Open_eVision_2_10.ERunData run
)
```

Parameters

run

Pointer to an [ERunData](#) to receive the data.

ECodedImage.GetLastRunPtr

Pointer to the last run list item, or **NULL**.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EListItem GetLastRunPtr (
)
```

ECodedImage.GetNextHole

Returns a pointer to the hole following the specified hole, in the objects list.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EListItem GetNextHole(
    Euresys.Open_eVision_2_10.EListItem hole
)
```

Parameters

hole

Pointer to the hole list item, for which the following hole has to be pointed out.

Remarks

If there is no hole yet in the list, the [ECodedImage::GetNextHole](#) function returns **NULL**.

ECodedImage.GetNextObjData

Moves the cursor to the next object and returns the associated data.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetNextObjData(
    out Euresys.Open_eVision_2_10.EObjectData object_
)
```

Parameters

object_

Pointer to an [EObjectData](#) to receive the data.

ECodedImage.GetNextObjPtr

Returns a pointer to the next objects list item, or **NULL**.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EListItem GetNextObjPtr(
    Euresys.Open_eVision_2_10.EListItem listItem
)
```

Parameters

listItem

Pointer to the current objects list item.

ECodedImage.GetNextRunData

Moves the cursor to the next run and returns the associated data.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetNextRunData (
    out Euresys.Open_eVision_2_10.ERunData run
)

void GetNextRunData (
    out Euresys.Open_eVision_2_10.ERunData run,
    Euresys.Open_eVision_2_10.EListItem listItem
)
```

Parameters

run

Pointer to an [ERunData](#) to receive the data.

listItem

Pointer to the current run list item.

ECodedImage.GetNextRunPtr

Returns a pointer to the next run list item, or **NULL**.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EListItem GetNextRunPtr (
    Euresys.Open_eVision_2_10.EListItem listItem
)
```

Parameters

listItem

Pointer to the current run list item.

ECodedImage.GetNumHoles

Returns the number of holes related to the specified object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int GetNumHoles (
    Euresys.Open_eVision_2_10.EListItem object_
)
```

Parameters

object_

Pointer to the object list item whose holes have to be counted.

Remarks

By default, the parameter **object** is set to **NULL**, meaning that the function returns the total number of holes added to the objects list. After a call to the [ECodedImage::BuildHoles](#) function, the [ECodedImage::NumObjects](#) and [ECodedImage::NumSelectedObjects](#) properties contain the total number of objects (i.e. real objects + holes).

ECodedImage.GetNumObjectRuns

Returns the number of runs comprised in a given object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int GetNumObjectRuns (
    int objectNumber
)

int GetNumObjectRuns (
    Euresys.Open_eVision_2_10.EListItem listItem
)
```

Parameters

objectNumber

Object identification number.

listItem

Pointer to an objects list item.

ECodedImage.GetObjDataPtr

Gets the [EObjectData](#) associated to a given objects list item.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool GetObjDataPtr(
    Euresys.Open_eVision_2_10.EListItem currentFeature,
    out Euresys.Open_eVision_2_10.EObjectData objectData
)
```

Parameters

currentFeature

Pointer to the current objects list item.

objectData

Pointer to a [EObjectData](#) to receive the data.

ECodedImage.GetObjectData

If an object identification number is given, moves the cursor to the object with a given identification number and returns the associated data. If a list item is given, returns the object data associated with an object list item (cf. [EListItem](#)). See also [ECodedImage::GetCurrentObjData](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetObjectData(
    out Euresys.Open_eVision_2_10.EObjectData object_,
    int objectNumber
)

void GetObjectData(
    out Euresys.Open_eVision_2_10.EObjectData object_,
    Euresys.Open_eVision_2_10.EListItem listItem
)
```

Parameters

object_

Pointer to an [EObjectData](#) structure to receive the object data.

objectNumber

Object identification number.

listItem

Pointer to the current object list item (cf. [EListItem](#)).

ECodedImage.GetObjectFeature

Allows retrieving the value of a feature of a given object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetObjectFeature(
    Euresys.Open_eVision_2_10.ELegacyFeature feature,
    Euresys.Open_eVision_2_10.EListItem objectNumber,
    out sbyte result
)

void GetObjectFeature(
    Euresys.Open_eVision_2_10.ELegacyFeature feature,
    Euresys.Open_eVision_2_10.EListItem object_,
    out short result
)

void GetObjectFeature(
    Euresys.Open_eVision_2_10.ELegacyFeature feature,
    Euresys.Open_eVision_2_10.EListItem object_,
    out int result
)

void GetObjectFeature(
    Euresys.Open_eVision_2_10.ELegacyFeature feature,
    Euresys.Open_eVision_2_10.EListItem object_,
    out float result
)

void GetObjectFeature(
    Euresys.Open_eVision_2_10.ELegacyFeature feature,
    Euresys.Open_eVision_2_10.EListItem object_,
    out double result
)
```

```
void GetObjectFeature(  
    int feature,  
    int objectNumber,  
    out sbyte result  
)  
  
void GetObjectFeature(  
    int feature,  
    int objectNumber,  
    out short result  
)  
  
void GetObjectFeature(  
    int feature,  
    int objectNumber,  
    out int result  
)  
  
void GetObjectFeature(  
    int feature,  
    int objectNumber,  
    out float result  
)  
  
void GetObjectFeature(  
    int feature,  
    int objectNumber,  
    out double result  
)  
  
void GetObjectFeature(  
    Euresys.Open_eVision_2_10.EListItem feature,  
    int objectNumber,  
    out sbyte result  
)  
  
void GetObjectFeature(  
    Euresys.Open_eVision_2_10.EListItem feature,  
    int objectNumber,  
    out short result  
)  
  
void GetObjectFeature(  
    Euresys.Open_eVision_2_10.EListItem feature,  
    int objectNumber,  
    out int result  
)
```



```
void GetObjectFeature(  
    Euresys.Open_eVision_2_10.EListItem feature,  
    int objectNumber,  
    out float result  
)  
  
void GetObjectFeature(  
    Euresys.Open_eVision_2_10.EListItem feature,  
    int objectNumber,  
    out double result  
)
```

Parameters

feature

Pointer to the feature list item.

objectNumber

Object number.

result

Reference to the feature value.

object_

Pointer to the list item (from the objects list) corresponding to the object.

ECodedImage.GetObjectFirstRunPtr

Returns a pointer to the first run list item of an object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
Euresys.Open_eVision_2_10.EListItem GetObjFirstRunPtr(  
    int objectNumber  
)  
  
Euresys.Open_eVision_2_10.EListItem GetObjFirstRunPtr(  
    Euresys.Open_eVision_2_10.EListItem listItem  
)
```

Parameters

objectNumber

Object identification number.

listItem

Pointer to the objects list item.

ECodedImage.GetObjLastRunPtr

Returns a pointer to the last run list item of an object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EListItem GetObjLastRunPtr(
    int objectNumber
)
Euresys.Open_eVision_2_10.EListItem GetObjLastRunPtr(
    Euresys.Open_eVision_2_10.EListItem objectNumber
)
```

Parameters

objectNumber

Object identification number.

ECodedImage.GetObjPtr

Returns a pointer to the given objects list item.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EListItem GetObjPtr(
    int objectNumber
)
```

Parameters

objectNumber

Object identification number.

ECodedImage.GetObjPtrByCoordinates

Returns a pointer to the objects list item that contains the point of given coordinates, or **NULL**.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EListItem GetObjPtrByCoordinates (
    int x,
    int y
)
```

Parameters

x

Point abscissa.

y

Point ordinate.

Remarks

This function is useful for object selection with a mouse.

ECodedImage.GetObjPtrByPos

Returns a pointer to the objects list item of given absolute position, or **NULL**.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EListItem GetObjPtrByPos (
    int position
)
```

Parameters

position

Absolute position in the objects list, counting from **0** on.

ECodedImage.GetPreviousObjData

Moves the cursor to the previous object and returns the associated data.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetPreviousObjData (
    out Euresys.Open_eVision_2_10.EObjectData object_
)
```

Parameters

object_

Pointer to an [EObjectData](#) to receive the data.

ECodedImage.GetPreviousObjPtr

Returns a pointer to the previous objects list item, or **NULL**.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EListItem GetPreviousObjPtr(
    Euresys.Open_eVision_2_10.EListItem listItem
)
```

Parameters

listItem

Pointer to the current objects list item.

ECodedImage.GetPreviousRunData

Moves the cursor to the previous run and returns the associated data.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetPreviousRunData (
    out Euresys.Open_eVision_2_10.ERunData run,
    Euresys.Open_eVision_2_10.EListItem listItem
)

void GetPreviousRunData (
    out Euresys.Open_eVision_2_10.ERunData run
)
```

Parameters

run

Pointer to an [ERunData](#) to receive the data.

listItem

Pointer to the current run list item.

ECodedImage.GetPreviousRunPtr

Returns a pointer to the previous run list item, or **NULL**.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EListItem GetPreviousRunPtr (
    Euresys.Open_eVision_2_10.EListItem listItem
)
```

Parameters

listItem

Pointer to the current run list item.

ECodedImage.GetRunData

Returns the run data associated to the specified run.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetRunData (
    out Euresys.Open_eVision_2_10.ERunData run,
    int position
)
```

```
void GetRunData (
    out Euresys.Open_eVision_2_10.ERunData run,
    Euresys.Open_eVision_2_10.EListItem listItem
)
```

Parameters

run

Pointer to an [ERunData](#) to receive the data.

position

Absolute position in the run list, counting from **0** on.

listItem

Pointer to the current run list item.

ECodedImage.GetRunDataPtr

Gets the [ERunData](#) associated to a given run list item.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool GetRunDataPtr (
    Euresys.Open_eVision_2_10.EListItem currentFeature,
    out Euresys.Open_eVision_2_10.ERunData runData
)
```

Parameters

currentFeature

Pointer to the current run list item.

runData

Pointer to a [ERunData](#) to receive the data.

ECodedImage.GetRunPtr

Returns a pointer to the run list item of given absolute position, or **NULL**.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EListItem GetRunPtr(
    int position
)
```

Parameters

position

Absolute position in the run list, counting from 0 on.

ECodedImage.GetRunPtrByCoordinates

Returns a pointer to the run list item that contains the point of given coordinates, or **NULL**.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EListItem GetRunPtrByCoordinates(
    int x,
    int y
)
```

Parameters

x

Point abscissa.

y

Point ordinate.

Remarks

This function is useful for run selection with a mouse.

ECodedImage.HighColorThreshold

Upper threshold (color) used for image segmentation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EC24 HighColorThreshold  
  
{ get; set; }
```

Remarks

The threshold value is constant over the whole image.

ECodedImage.HighImage

Image used as an adaptive upper threshold.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EROIBW8 HighImage  
  
{ get; set; }
```

Remarks

The threshold is adaptive (specified pixel by pixel).

ECodedImage.HighThreshold

Upper threshold (gray level) used for image segmentation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint HighThreshold  
  
    { get; set; }
```

Remarks

The threshold value is constant over the whole image.

ECodedImage.IsHole

Returns **TRUE** if the specified object is a hole, **FALSE** otherwise.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool IsHole (  
    Euresys.Open_eVision_2_10.EListItem object_  
)
```

Parameters

object_
Pointer to the objects list item.

ECodedImage.IsObjectSelected

Sets **bSelected** to **TRUE** if an object is selected.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void IsObjectSelected(
    int objectNumber,
    out bool selected
)

void IsObjectSelected(
    Euresys.Open_eVision_2_10.EListItem listItem,
    out bool selected
)
```

Parameters

objectNumber

Object identification number.

selected

Reference to the result.

listItem

Pointer to the objects list item.

Remarks

ECodedImage.LastObjPtr

Pointer to the last objects list item, or **NULL**.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EListItem LastObjPtr
{ get; }
```

ECodedImage.LimitAngle

Angle of the skewed bounding box feature, in the current angle unit.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float LimitAngle
{ get; set; }
```

ECodedImage.LowColorThreshold

Lower threshold (color) used for image segmentation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EC24 LowColorThreshold
{ get; set; }
```

Remarks

The threshold value is constant over the whole image.

ECodedImage.LowImage

Image used as an adaptive lower threshold.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EROIBW8 LowImage  
  
{ get; set; }
```

Remarks

The threshold value is adaptive (specified pixel by pixel).

ECodedImage.LowThreshold

Lower threshold (gray level) used for image segmentation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint LowThreshold  
  
{ get; set; }
```

Remarks

The threshold value is constant over the whole image.

ECodedImage.MaxObjects

Maximum number of objects to look for.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint MaxObjects  
  
{ get; set; }
```

Remarks

After having found that amount of object, the process will stop and return an error message. If not set, no maximum value is defined.

ECodedImage.NeutralClass

Neutral class index (between both thresholds).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
short NeutralClass  
  
{ get; set; }
```

Remarks

Non zero when the neutral runs (between both thresholds) are coded. **0** means "do not code this class". <!-- **2** by default. -->

ECodedImage.NumFeatures

Number of features currently in use.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int NumFeatures  
    { get; }
```

ECodedImage.NumHoleRuns

Total number of hole runs in the list of object runs.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int NumHoleRuns  
    { get; }
```

Remarks

After a call to [ECodedImage::BuildHoles](#), [ECodedImage::NumRuns](#) contains the total number of runs (real object runs + hole runs).

ECodedImage.NumObjects

Number of objects in the coded image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int NumObjects
{ get; }
```

Remarks

After a call to [ECodedImage::BuildHoles](#), [ECodedImage::NumObjects](#) returns the total number of objects (real objects + holes).

ECodedImage.NumRuns

Total number of runs in the list of object runs.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int NumRuns
{ get; }
```

Remarks

After a call to [ECodedImage::BuildHoles](#), [ECodedImage::NumRuns](#) returns the total number of runs (real object runs + hole runs).

ECodedImage.NumSelectedObjects

Number of objects currently selected.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
int NumSelectedObjects
{ get; set; }
```

Remarks

After a call to [ECodedImage::BuildHoles](#), [ECodedImage::NumSelectedObjects](#) returns the total number of objects (real objects + holes).

ECodedImage.ObjectConvexHull

Computes the convex hull of an object and stores it in a **EPathVector**.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ObjectConvexHull(
    Euresys.Open_eVision_2_10.EListItem object_,
    Euresys.Open_eVision_2_10.EPathVector destinationVector
)
```

Parameters

object_

Pointer to the objects list item.

destinationVector

Vector containing the vertices coordinates of the convex hull.

ECodedImage.RemoveAllFeats

Deletes all features from the features list.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void RemoveAllFeats (  
)
```

ECodedImage.RemoveAllObjects

Deletes all objects from the objects list.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void RemoveAllObjects (  
)
```

ECodedImage.RemoveAllRuns

Deletes all runs from the runs list.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void RemoveAllRuns (  
)
```

ECodedImage.RemoveHoles

Permanently erases, from the objects list, holes related to the specified object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void RemoveHoles (
    Euresys.Open_eVision_2_10.EListItem object_
)
```

Parameters

object_

Pointer to the objects list item, for which the holes have to be erased.

Remarks

If the parameter is **NULL**, all the holes are deleted. By default, the parameter is set to **NULL**, meaning that all the holes have to be erased from the objects list.

ECodedImage.RemoveObject

Deletes an object from the objects list.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void RemoveObject (
    int objectNumber
)

void RemoveObject (
    Euresys.Open_eVision_2_10.EListItem listItem
)
```

Parameters

objectNumber

Object identification number.

listItem

Pointer to the current objects list item.

ECodedImage.RemoveRun

Deletes a run from the runs list.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void RemoveRun (
    int position
)
void RemoveRun (
    Euresys.Open_eVision_2_10.EListItem listItem
)
```

Parameters

position

Absolute position in the run list, counting from 0 on.

listItem

Pointer to the current run list item.

ECodedImage.ResetContinuousMode

When the continuous mode is activated, this method resets the sequence of images.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void ResetContinuousMode (  
)
```

Remarks

Thus, the next call for an object building will not take into account any previous image. If the continuous mode is disabled, this method does nothing.

ECodedImage.SelectAllObjects

Selects all objects.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void SelectAllObjects (  
)
```

ECodedImage.SelectHoles

Selects the holes related to the specified object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void SelectHoles (  
    Euresys.Open_eVision_2_10.EListItem parentObject  
)
```

Parameters

parentObject

Pointer to the objects list item, for which the holes have to be selected.

Remarks

If the parameter is **NULL**, all the holes are selected. By default, the parameter is set to **NULL**, meaning that all the holes have to be selected. If **parentObject** is a hole (instead of a real object) or has no hole, no selection is performed.

ECodedImage.SelectObject

Selects an object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void SelectObject(  
    int objectNumber  
)  
  
void SelectObject(  
    Euresys.Open_eVision_2_10.EListItem listItem  
)
```

Parameters

objectNumber

Object identification number.

listItem

Pointer to the objects list item.

ECodedImage.SelectObjectsUsingFeature

Selects or deselects objects, according to the value of a specified feature.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void SelectObjectsUsingFeature(
    Euresys.Open_eVision_2_10.ELegacyFeature feature,
    sbyte minimum,
    sbyte maximum,
    Euresys.Open_eVision_2_10.ESelectOption operation
)

void SelectObjectsUsingFeature(
    Euresys.Open_eVision_2_10.ELegacyFeature feature,
    short minimum,
    short maximum,
    Euresys.Open_eVision_2_10.ESelectOption operation
)

void SelectObjectsUsingFeature(
    Euresys.Open_eVision_2_10.ELegacyFeature feature,
    int minimum,
    int maximum,
    Euresys.Open_eVision_2_10.ESelectOption operation
)

void SelectObjectsUsingFeature(
    Euresys.Open_eVision_2_10.ELegacyFeature feature,
    float minimum,
    float maximum,
    Euresys.Open_eVision_2_10.ESelectOption operation
)

void SelectObjectsUsingFeature(
    Euresys.Open_eVision_2_10.ELegacyFeature feature,
    double minimum,
    double maximum,
    Euresys.Open_eVision_2_10.ESelectOption operation
)
```

Parameters

feature

Feature code, as defined by [EFeature](#).

minimum

Selection interval lower bound.

maximum

Selection interval upper bound.

operation

Selection mode, as defined by [ESelectOption](#).

ECodedImage.SelectObjectsUsingPosition

Selects or deselects objects, using a delimiting rectangle.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void SelectObjectsUsingPosition(  
    Euresys.Open_eVision_2_10.EBaseROI roi,  
    Euresys.Open_eVision_2_10.ESelectByPosition operation  
)  
  
void SelectObjectsUsingPosition(  
    int originX,  
    int originY,  
    int width,  
    int height,  
    Euresys.Open_eVision_2_10.ESelectByPosition operation  
)
```

Parameters

roi

Pointer to an image/ROI whose position parameters will define the selection rectangle.

operation

Selection mode, as defined by [ESelectByPosition](#).

originX

Abscissa of the upper left corner of the rectangle.

originY

Ordinate of the upper left corner of the rectangle.

width

Rectangle width, in pixels.

height

Rectangle height, in pixels.

Remarks

The rectangle coordinates are always specified with respect to the whole image.

ECodedImage.SetFeatInfo

Sets the appropriate data size and type for a predefined feature.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetFeatInfo(
    ref Euresys.Open_eVision_2_10.EFeatureData feature,
    Euresys.Open_eVision_2_10.ELegacyFeature featureCode
)
```

Parameters

feature

Feature code, as defined by [ELegacyFeature](#).

featureCode

Pointer to a [EFeatureData](#) structure describing the feature.

ECodedImage.SetFirstRunPtr

Sets the first run list item of an object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetFirstRunPtr(
    Euresys.Open_eVision_2_10.EListItem firstRun,
    int objectNumber
)

void SetFirstRunPtr(
    Euresys.Open_eVision_2_10.EListItem firstRun,
    Euresys.Open_eVision_2_10.EListItem currentObject
)
```

Parameters

firstRun

Pointer to the first run of the object.

objectNumber

Object identification number.

currentObject

Pointer to the objects list item.

ECodedImage.SetLastRunPtr

Sets the last run list item of an object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetLastRunPtr(
    Euresys.Open_eVision_2_10.EListItem lastRun,
    int objectNumber
)

void SetLastRunPtr(
    Euresys.Open_eVision_2_10.EListItem lastRun,
    Euresys.Open_eVision_2_10.EListItem currentObject
)
```

Parameters

lastRun

Pointer to the last run of the object.

objectNumber

Object identification number.

currentObject

Pointer to the objects list item.

ECodedImage.SortObjectsUsingFeature

Sorts objects according to the value of some feature.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SortObjectsUsingFeature (
    Euresys.Open_eVision_2_10.ELegacyFeature feature,
    Euresys.Open_eVision_2_10.ESortOption Operation
)
```

Parameters

feature

Feature code, as defined by [ELegacyFeature](#).

Operation

Selection mode, as defined by [ESortOption](#).

ECodedImage.Threshold

Threshold mode (gray level) used for image segmentation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint Threshold
{ get; set; }
```

Remarks

By default, the "minimum residue" mode is used to determine the threshold. The threshold is constant over the whole image. When using a single threshold instead of a low threshold and a high threshold, the neutral class is ignored. Only the black and white classes are relevant.

ECodedImage.ThresholdImage

Single threshold used for image segmentation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EROIBW8 ThresholdImage
{ get; set; }
```

Remarks

The threshold value is adaptive (specified pixel by pixel). When using a single threshold instead of a low threshold and a high threshold, the neutral class is ignored. Only the black and white classes are relevant.

ECodedImage.TrueThreshold

Absolute threshold level, when using a single threshold.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint TrueThreshold
{ get; }
```

ECodedImage.UnselectAllObjects

Deselects all objects.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void UnselectAllObjects (
)
```

ECodedImage.UnselectHoles

Unselects the holes related to the specified object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void UnselectHoles (
    Euresys.Open_eVision_2_10.EListItem parentObject
)
```

Parameters

parentObject

Pointer to the objects list item, for which the holes have to be unselected.

Remarks

If the parameter is **NULL**, all the holes are unselected. By default, the parameter is set to **NULL**, meaning that all the holes have to be unselected. If **parentObject** is a hole (instead of a real object) or if **parentObject** has no hole, nothing is changed.

ECodedImage.UnselectObject

Deselects an object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void UnselectObject(
    int objectNumber
)

void UnselectObject(
    Euresys.Open_eVision_2_10.EListItem listItem
)
```

Parameters

objectNumber

Object identification number.

listItem

Pointer to the objects list item.

Remarks

Once an object has been unselected, it doesn't allow browsing a list of selected objects anymore (using [ECodedImage::GetPreviousObjPtr](#) or [ECodedImage::GetNextObjPtr](#)).

ECodedImage.WhiteClass

White class index (above the upper threshold).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
short WhiteClass  
  
    { get; set; }
```

Remarks

Non zero when the white runs (above the upper threshold) are coded. **0** means "do not code this class". <!-- **3** by default. -->

4.48. ECodedImage2 Class

The main class of the EasyObject API that represents a coded image, as produced by the encoder.

Remarks

It provides methods to get features of the encoded image, to access an object in a particular layer of the encoded image, to draw objects or objects features in a particular layer of the encoded image, to render a mask, etc...

Namespace: Euresys.Open_eVision_2_10

Properties

Height

Returns the height of the coded image.

LayerCount

Returns the number of layers that are encoded.

StartY

Returns the lowest row index, for all the runs of all the objects in the coded image.

Width

Returns the width of the coded image.

Methods

ClearFeatureCache

Clears the internal cache for the computed features.

Draw

Draw the designated coded element, all the objects from a layer, or all the coded elements from a given selection.

DrawFeature

Draw a given feature of the designated coded element, of all the objects from a layer, or of all the coded elements from a given selection.

DrawFeatureWithCurrentPen

Draw a given feature of the designated coded element, of all the objects from a layer, or of all the coded elements from a given selection.

DrawHole

Draw the designated hole.

DrawHoleFeature

Draw a given feature of the designated hole.

DrawHoleFeatureWithCurrentPen

Draw a given feature of the designated hole.

DrawHoleWithCurrentPen

Draw the designated hole.

DrawObject

Draw the designated object.

DrawObjectFeature

Draw a given feature of the designated object.

DrawObjectFeatureWithCurrentPen

Draw a given feature of the designated object.

DrawObjectWithCurrentPen

Draw the designated object.

DrawWithCurrentPen

Draw the designated coded element, all the objects from a layer, or all the coded elements from a given selection.

ECodedImage2

Constructs a coded image.

FindObject

Finds an object in the coded image using its coordinates.

GetObj

Random access to an object in a given layer.

GetObjCount

Returns the number of objects in a given layer.

GetParentObject

Returns a reference to the object that contains a given hole.

RenderMask

Creates a Flexible Mask from a specified layer of the encoded image.

C

odedImage2.ClearFeatureCache

Clears the internal cache for the computed features.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void ClearFeatureCache(  
    )
```

Remarks

This is useful to reduce memory consumption.

ECodedImage2.Draw

Draw the designated coded element, all the objects from a layer, or all the coded elements from a given selection.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.ECodedElement element,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
    )
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
    )
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    uint layerIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EObjectSelection selection,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EObjectSelection selection,  
    uint elementIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ECodedElement element,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ERGBColor color,  
    Euresys.Open_eVision_2_10.ECodedElement element,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ERGBColor color,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    uint layerIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ERGBColor color,  
    uint layerIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EObjectSelection selection,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ERGBColor color,  
    Euresys.Open_eVision_2_10.EObjectSelection selection,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EObjectSelection selection,  
    uint elementIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ERGBColor color,  
    Euresys.Open_eVision_2_10.EObjectSelection selection,  
    uint elementIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

Parameters

graphicContext

Graphic context on which to draw.

element

The coded element to draw.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

layerIndex

Index of the layer of interest. If no layer index is specified, the default layer is taken into consideration. Note that this methods throws an exception if no layer index is specified and if, simultaneously, the coded image contains several layers. Indeed, in such a case, no default layer exists.

selection

The selection of coded elements to draw.

elementIndex

The index in the selection of the coded element to draw. If no element index is specified, all the elements of the selection are take into consideration.

color

The color in which to draw the overlay.

Remarks

Drawing is done in the device context associated to the desired window. The current pen is used.

ECodedImage2.DrawFeature

Draw a given feature of the designated coded element, of all the objects from a layer, or of all the coded elements from a given selection.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void DrawFeature(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EDrawableFeature feature,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)
```

```
void DrawFeature(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EDrawableFeature feature,  
    uint layerIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)
```

```
void DrawFeature(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EDrawableFeature feature,  
    Euresys.Open_eVision_2_10.ECodedElement element,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)
```

```
void DrawFeature(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EDrawableFeature feature,  
    Euresys.Open_eVision_2_10.EObjectSelection selection,  
    uint elementIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)
```

```
void DrawFeature(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EDrawableFeature feature,  
    Euresys.Open_eVision_2_10.EObjectSelection selection,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)
```

```
void DrawFeature(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EDrawableFeature feature,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)
```

```
void DrawFeature(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ERGBColor color,  
    Euresys.Open_eVision_2_10.EDrawableFeature feature,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)
```

```
void DrawFeature(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EDrawableFeature feature,  
    uint layerIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)
```



```
void DrawFeature(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ERGBColor color,  
    Euresys.Open_eVision_2_10.EDrawableFeature feature,  
    uint layerIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)  
  
void DrawFeature(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EDrawableFeature feature,  
    Euresys.Open_eVision_2_10.ECodedElement element,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)  
  
void DrawFeature(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ERGBColor color,  
    Euresys.Open_eVision_2_10.EDrawableFeature feature,  
    Euresys.Open_eVision_2_10.ECodedElement element,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)  
  
void DrawFeature(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EDrawableFeature feature,  
    Euresys.Open_eVision_2_10.EObjectSelection selection,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)
```

```
void DrawFeature(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ERGBColor color,  
    Euresys.Open_eVision_2_10.EDrawableFeature feature,  
    Euresys.Open_eVision_2_10.EObjectSelection selection,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)
```

```
void DrawFeature(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EDrawableFeature feature,  
    Euresys.Open_eVision_2_10.EObjectSelection selection,  
    uint elementIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)
```

```
void DrawFeature(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ERGBColor color,  
    Euresys.Open_eVision_2_10.EDrawableFeature feature,  
    Euresys.Open_eVision_2_10.EObjectSelection selection,  
    uint elementIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)
```

Parameters

graphicContext

Graphic context on which to draw.

feature

The feature of interest.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

drawDiagonals

Specifies whether or not the diagonals of the ellipses and of the rectangles are to be drawn.

layerIndex

Index of the layer of interest. If no layer index is specified, the default layer is taken into consideration. Note that this methods throws an exception if no layer index is specified and if, simultaneously, the coded image contains several layers. Indeed, in such a case, no default layer exists.

element

The coded element to draw.

selection

The selection of coded elements to draw.

elementIndex

The index in the selection of the coded element to draw. If no element index is specified, all the elements of the selection are take into consideration.

color

The color in which to draw the overlay.

Remarks

Drawing is done in the device context associated to the desired window. The current pen is used.

The features [FerretBox](#) and [WeightedGravityCenter](#) are only at one's disposal when drawing selections.

ECodedImage2.DrawFeatureWithCurrentPen

Draw a given feature of the designated coded element, of all the objects from a layer, or of all the coded elements from a given selection.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void DrawFeatureWithCurrentPen(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EDrawableFeature feature,  
    uint layerIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)  
  
void DrawFeatureWithCurrentPen(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EDrawableFeature feature,  
    Euresys.Open_eVision_2_10.ECodedElement element,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)  
  
void DrawFeatureWithCurrentPen(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EDrawableFeature feature,  
    Euresys.Open_eVision_2_10.EObjectSelection selection,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)  
  
void DrawFeatureWithCurrentPen(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EDrawableFeature feature,  
    Euresys.Open_eVision_2_10.EObjectSelection selection,  
    uint elementIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)
```

```
void DrawFeatureWithCurrentPen(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EDrawableFeature feature,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)
```

Parameters

graphicContext

Graphic context on which to draw.

feature

The feature of interest.

layerIndex

Index of the layer of interest. If no layer index is specified, the default layer is taken into consideration. Note that this methods throws an exception if no layer index is specified and if, simultaneously, the coded image contains several layers. Indeed, in such a case, no default layer exists.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

drawDiagonals

Specifies whether or not the diagonals of the ellipses and of the rectangles are to be drawn.

element

The coded element to draw.

selection

The selection of coded elements to draw.

elementIndex

The index in the selection of the coded element to draw. If no element index is specified, all the elements of the selection are take into consideration.

Remarks

Drawing is done in the device context associated to the desired window. The current pen is used.

The features [FeretBox](#) and [WeightedGravityCenter](#) are only at one's disposal when drawing selections.

ECodedImage2.DrawHole

Draw the designated hole.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void DrawHole(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    uint objectIndex,  
    uint holeIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void DrawHole(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    uint layerIndex,  
    uint objectIndex,  
    uint holeIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void DrawHole(  
    IntPtr graphicContext,  
    uint objectIndex,  
    uint holeIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void DrawHole(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ERGBColor color,  
    uint objectIndex,  
    uint holeIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void DrawHole(  
    IntPtr graphicContext,  
    uint layerIndex,  
    uint objectIndex,  
    uint holeIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void DrawHole(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ERGBColor color,  
    uint layerIndex,  
    uint objectIndex,  
    uint holeIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

Parameters

graphicContext

Graphic context on which to draw.

objectIndex

Index of the parent object of the hole to draw.

holeIndex

Index of the hole to draw.

zoomX

Horizontal zooming factor. By default, no scaling is used.

zoomY

Vertical zooming factor. If set to **0**, the default value, the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

layerIndex

The index of the layer of interest. If this parameter is left unspecified, the default layer will be taken into consideration.

color

The color in which to draw the overlay.

Remarks

Drawing is done in the device context associated to the desired window.

This methods throws an exception if no layer index is specified and if, simultaneously, the coded image contains several layers. Indeed, in such a case, no default layer exists.

ECodedImage2.DrawHoleFeature

Draw a given feature of the designated hole.

Namespace: Euresys.Open_eVision_2_10

[C#]


```
void DrawHoleFeature(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EDrawableFeature feature,  
    uint objectIndex,  
    uint holeIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)
```

```
void DrawHoleFeature(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EDrawableFeature feature,  
    uint layerIndex,  
    uint objectIndex,  
    uint holeIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)
```

```
void DrawHoleFeature(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EDrawableFeature feature,  
    uint objectIndex,  
    uint holeIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)
```

```

void DrawHoleFeature(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.EDrawableFeature feature,
    uint objectIndex,
    uint holeIndex,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    bool drawDiagonals
)

void DrawHoleFeature(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawableFeature feature,
    uint layerIndex,
    uint objectIndex,
    uint holeIndex,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    bool drawDiagonals
)

void DrawHoleFeature(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.EDrawableFeature feature,
    uint layerIndex,
    uint objectIndex,
    uint holeIndex,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    bool drawDiagonals
)

```

Parameters

graphicContext

Graphic context on which to draw.

feature

The feature of interest.

objectIndex

Index of the parent object of the hole to draw.

holeIndex

Index of the hole to draw.

zoomX

Horizontal zooming factor. By default, no scaling is used.

zoomY

Vertical zooming factor. If set to **0**, the default value, the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

drawDiagonals

Specifies whether or not the diagonals of the ellipses and of the rectangles are to be drawn.

layerIndex

The index of the layer of interest. If this parameter is left unspecified, the default layer will be taken into consideration.

color

The color in which to draw the overlay.

Remarks

Drawing is done in the device context associated to the desired window. The current pen is used.

This methods throws an exception if no layer index is specified and if, simultaneously, the coded image contains several layers. Indeed, in such a case, no default layer exists.

Trying to draw the features [FeretBox](#) and [WeightedGravityCenter](#) will result in an exception, as they only make sense for [EObjectSelection](#).

ECodedImage2.DrawHoleFeatureWithCurrentPen

Draw a given feature of the designated hole.

Namespace: Euresys.Open_eVision_2_10

```

[C#]
void DrawHoleFeatureWithCurrentPen(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawableFeature feature,
    uint objectIndex,
    uint holeIndex,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    bool drawDiagonals
)

void DrawHoleFeatureWithCurrentPen(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawableFeature feature,
    uint layerIndex,
    uint objectIndex,
    uint holeIndex,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    bool drawDiagonals
)

```

Parameters

graphicContext

Graphic context on which to draw.

feature

The feature of interest.

objectIndex

Index of the parent object of the hole to draw.

holeIndex

Index of the hole to draw.

zoomX

Horizontal zooming factor. By default, no scaling is used.

zoomY

Vertical zooming factor. If set to **0**, the default value, the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

drawDiagonals

Specifies whether or not the diagonals of the ellipses and of the rectangles are to be drawn.

layerIndex

The index of the layer of interest. If this parameter is left unspecified, the default layer will be taken into consideration.

Remarks

Drawing is done in the device context associated to the desired window. The current pen is used.

This methods throws an exception if no layer index is specified and if, simultaneously, the coded image contains several layers. Indeed, in such a case, no default layer exists.

Trying to draw the features [FerretBox](#) and [WeightedGravityCenter](#) will result in an exception, as they only make sense for [EObjectSelection](#).

ECodedImage2.DrawHoleWithCurrentPen

Draw the designated hole.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawHoleWithCurrentPen(
    IntPtr graphicContext,
    uint layerIndex,
    uint objectIndex,
    uint holeIndex,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

```
void DrawHoleWithCurrentPen(  
    IntPtr graphicContext,  
    uint objectIndex,  
    uint holeIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

Parameters

graphicContext

Graphic context on which to draw.

layerIndex

The index of the layer of interest. If this parameter is left unspecified, the default layer will be taken into consideration.

objectIndex

Index of the parent object of the hole to draw.

holeIndex

Index of the hole to draw.

zoomX

Horizontal zooming factor. By default, no scaling is used.

zoomY

Vertical zooming factor. If set to **0**, the default value, the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

Remarks

Drawing is done in the device context associated to the desired window. The current pen is used.

This methods throws an exception if no layer index is specified and if, simultaneously, the coded image contains several layers. Indeed, in such a case, no default layer exists.

ECodedImage2.DrawObject

Draw the designated object.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void DrawObject(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    uint objectIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void DrawObject(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    uint layerIndex,  
    uint objectIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void DrawObject(  
    IntPtr graphicContext,  
    uint objectIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```

void DrawObject(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    uint objectIndex,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void DrawObject(
    IntPtr graphicContext,
    uint layerIndex,
    uint objectIndex,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void DrawObject(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    uint layerIndex,
    uint objectIndex,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

```

Parameters

graphicContext

Graphic context on which to draw.

objectIndex

Index of the object to draw.

zoomX

Horizontal zooming factor. By default, no scaling is used.

zoomY

Vertical zooming factor. If set to **0**, the default value, the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

layerIndex

The index of the layer of interest. If this parameter is left unspecified, the default layer will be taken into consideration.

color

The color in which to draw the overlay.

Remarks

Drawing is done in the device context associated to the desired window. The current pen is used.

This methods throws an exception if no layer index is specified and if, simultaneously, the coded image contains several layers. Indeed, in such a case, no default layer exists.

ECodedImage2.DrawObjectFeature

Draw a given feature of the designated object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawObjectFeature (
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.EDrawableFeature feature,
    uint objectIndex,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    bool drawDiagonals
)
```

```
void DrawObjectFeature(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EDrawableFeature feature,  
    uint layerIndex,  
    uint objectIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)
```

```
void DrawObjectFeature(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EDrawableFeature feature,  
    uint objectIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)
```

```
void DrawObjectFeature(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ERGBColor color,  
    Euresys.Open_eVision_2_10.EDrawableFeature feature,  
    uint objectIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)
```

```

void DrawObjectFeature(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawableFeature feature,
    uint layerIndex,
    uint objectIndex,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    bool drawDiagonals
)

void DrawObjectFeature(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.EDrawableFeature feature,
    uint layerIndex,
    uint objectIndex,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    bool drawDiagonals
)

```

Parameters

graphicContext

Graphic context on which to draw.

feature

The feature of interest.

objectIndex

Index of the object to draw.

zoomX

Horizontal zooming factor. By default, no scaling is used.

zoomY

Vertical zooming factor. If set to **0**, the default value, the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

drawDiagonals

Specifies whether or not the diagonals of the ellipses and of the rectangles are to be drawn.

layerIndex

The index of the layer of interest. If this parameter is left unspecified, the default layer will be taken into consideration.

color

The color in which to draw the overlay.

Remarks

Drawing is done in the device context associated to the desired window. The current pen is used.

This methods throws an exception if no layer index is specified and if, simultaneously, the coded image contains several layers. Indeed, in such a case, no default layer exists.

Trying to draw the features [FeretBox](#) and [WeightedGravityCenter](#) will result in an exception, as they only make sense for [EObjectSelection](#).

ECodedImage2.DrawObjectFeatureWithCurrentPen

Draw a given feature of the designated object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawObjectFeatureWithCurrentPen(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawableFeature feature,
    uint objectIndex,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    bool drawDiagonals
)
```

```
void DrawObjectFeatureWithCurrentPen(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EDrawableFeature feature,  
    uint layerIndex,  
    uint objectIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)
```

Parameters

graphicContext

Graphic context on which to draw.

feature

The feature of interest.

objectIndex

Index of the object to draw.

zoomX

Horizontal zooming factor. By default, no scaling is used.

zoomY

Vertical zooming factor. If set to **0**, the default value, the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

drawDiagonals

Specifies whether or not the diagonals of the ellipses and of the rectangles are to be drawn.

layerIndex

The index of the layer of interest. If this parameter is left unspecified, the default layer will be taken into consideration.

Remarks

Drawing is done in the device context associated to the desired window. The current pen is used.

This methods throws an exception if no layer index is specified and if, simultaneously, the coded image contains several layers. Indeed, in such a case, no default layer exists.

Trying to draw the features [FerretBox](#) and [WeightedGravityCenter](#) will result in an exception, as they only make sense for [EObjectSelection](#).

ECodedImage2.DrawObjectWithCurrentPen

Draw the designated object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawObjectWithCurrentPen (
    IntPtr graphicContext,
    uint objectIndex,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void DrawObjectWithCurrentPen (
    IntPtr graphicContext,
    uint layerIndex,
    uint objectIndex,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

Parameters

graphicContext

Graphic context on which to draw.

objectIndex

Index of the object to draw.

zoomX

Horizontal zooming factor. By default, no scaling is used.

zoomY

Vertical zooming factor. If set to **0**, the default value, the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

layerIndex

The index of the layer of interest. If this parameter is left unspecified, the default layer will be taken into consideration.

Remarks

Drawing is done in the device context associated to the desired window. The current pen is used.

This methods throws an exception if no layer index is specified and if, simultaneously, the coded image contains several layers. Indeed, in such a case, no default layer exists.

ECodedImage2.DrawWithCurrentPen

Draw the designated coded element, all the objects from a layer, or all the coded elements from a given selection.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawWithCurrentPen(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ECodedElement element,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

```

void DrawWithCurrentPen(
    IntPtr graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void DrawWithCurrentPen(
    IntPtr graphicContext,
    uint layerIndex,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void DrawWithCurrentPen(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EObjectSelection selection,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void DrawWithCurrentPen(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EObjectSelection selection,
    uint elementIndex,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

```

Parameters

graphicContext

Graphic context on which to draw.

element

The coded element to draw.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

layerIndex

Index of the layer of interest. If no layer index is specified, the default layer is taken into consideration. Note that this methods throws an exception if no layer index is specified and if, simultaneously, the coded image contains several layers. Indeed, in such a case, no default layer exists.

selection

The selection of coded elements to draw.

elementIndex

The index in the selection of the coded element to draw. If no element index is specified, all the elements of the selection are take into consideration.

Remarks

Drawing is done in the device context associated to the desired window. The current pen is used.

ECodedImage2.ECodedImage2

Constructs a coded image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ECodedImage2 (
    Euresys.Open_eVision_2_10.ECodedImage2 other
)
void ECodedImage2 (
)
```

Parameters

other

-

ECodedImage2.FindObject

Finds an object in the coded image using its coordinates.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EObject FindObject(
    int x,
    int y
)

Euresys.Open_eVision_2_10.EObject FindObject(
    uint layerIndex,
    int x,
    int y
)
```

Parameters

x

The X-coordinate of the object.

y

The Y-coordinate of the object.

layerIndex

The index of the layer of interest.

Remarks

If no layer index is specified, all the layers of the coded image are scanned until an object is found at these coordinates.

ECodedImage2.GetObj

Random access to an object in a given layer.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EObject GetObj (
    uint layerIndex,
    uint objectIndex
)

Euresys.Open_eVision_2_10.EObject GetObj (
    uint layerIndex,
    uint objectIndex
)

Euresys.Open_eVision_2_10.EObject GetObj (
    uint objectIndex
)

Euresys.Open_eVision_2_10.EObject GetObj (
    uint objectIndex
)
```

Parameters

layerIndex

The index of the layer of interest. If this parameter is left unspecified, the default layer will be taken into consideration.

objectIndex

The index of the object in the layer.

Remarks

This methods throws an exception if no layer index is specified and if, simultaneously, the coded image contains several layers. Indeed, in such a case, no default layer exists.

ECodedImage2.GetObjCount

Returns the number of objects in a given layer.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint GetObjCount(  
    uint layerIndex  
)  
  
uint GetObjCount(  
)
```

Parameters

layerIndex

The index of the layer of interest. If this parameter is left unspecified, the default layer will be taken into consideration.

Remarks

This methods throws an exception if no layer index is specified and if, simultaneously, the coded image contains several layers. Indeed, in such a case, no default layer exists.

ECodedImage2.GetParentObject

Returns a reference to the object that contains a given hole.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
Euresys.Open_eVision_2_10.EObject GetParentObject(  
    Euresys.Open_eVision_2_10.EHole hole  
)
```

Parameters

hole

The hole of interest.

ECodedImage2.Height

Returns the height of the coded image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint Height  
    { get; }
```

Remarks

If the continuous mode is not activated, this height corresponds to the height of the source image. If the continuous mode is activated, this value equals to the highest row index, for all the runs of all the objects in the coded image, augmented by the number of rows index that are below zero.

ECodedImage2.LayerCount

Returns the number of layers that are encoded.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint LayerCount  
    { get; }
```

ECodedImage2.RenderMask

Creates a Flexible Mask from a specified layer of the encoded image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void RenderMask(
    Euresys.Open_eVision_2_10.EROIBW8 result
)

void RenderMask(
    Euresys.Open_eVision_2_10.EROIBW8 result,
    uint layerIndex,
    int offsetX,
    int offsetY
)

void RenderMask(
    Euresys.Open_eVision_2_10.EROIBW8 result,
    uint layerIndex
)

void RenderMask(
    Euresys.Open_eVision_2_10.EROIBW8 result,
    int offsetX,
    int offsetY
)
```

Parameters

result

The image in which the generated mask will be stored.

layerIndex

The index of the layer of interest. If this parameter is left unspecified, the default layer will serve as a source for the mask generation.

offsetX

The X-offset that must be applied to bring the zero X-coordinate in the coded image on the first column of the result image (defaults to zero).

offsetY

The Y-offset that must be applied to bring the zero Y-coordinate in the coded image on the first row of the result image (defaults to zero).

Remarks

The size of the result image will not be changed: It must be properly sized beforehand.

This methods throws an exception if no layer index is specified and if, simultaneously, the coded image contains several layers. Indeed, in such a case, no default layer exists.

ECodedImage2.StartY

Returns the lowest row index, for all the runs of all the objects in the coded image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int StartY  
  
{ get; }
```

Remarks

The returned value will always be zero if the continuous mode is not activated.

ECodedImage2.Width

Returns the width of the coded image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint Width  
  
{ get; }
```

Remarks

This width corresponds in any case to the width of the source image.

4.49. EColorLookup Class

Describes a color lookup table, that is used to speed-up complex conversions between color systems.

Namespace: Euresys.Open_eVision_2_10

Properties

ColorSystemIn

Input color system.

ColorSystemOut

Output color system.

IndexBits

Number of bits used for indexing the lookup table.

Interpolation

M Interpolation mode.

e

Methods

AdjustGainOffset

Sets a color transformation such that a gain and offset is applied separately to each color component of a target color system.

Calibrate

Sets a color transformation to recalibrate.

ConvertFromRgb

Sets a color transformation from the [Rgb](#) representation to another system, as defined by [EColorSystem](#).

ConvertToRgb

Sets a color transformation from any color system, as defined by [EColorSystem](#), to the [Rgb](#) representation.

EColorLookup

Constructs a color lookup table.

Transform

Transforms a quantized color image/pixel to another quantized color image/pixel, using the previously initialized color lookup table.

WhiteBalance

Initializes a color lookup table that can be used for color adjustment, including a global gain, gamma pre-compensation [cancellation] and white balancing.

ColorLookup.AdjustGainOffsets
et

Sets a color transformation such that a gain and offset is applied separately to each color component of a target color system.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void AdjustGainOffset(  
    Euresys.Open_eVision_2_10.EColorSystem colorSystem,  
    float gain0,  
    float offset0,  
    float gain1,  
    float offset1,  
    float gain2,  
    float offset2  
)
```

Parameters

colorSystem

Target color system, as defined by [EColorSystem](#).

gain0

Gain to be applied to color component 0.

offset0

Offset to be applied to color component 0.

gain1

Gain to be applied to color component 1.

offset1

Offset to be applied to color component 1.

gain2

Gain to be applied to color component 2.

offset2

Offset to be applied to color component 2.

Remarks

The input and output color systems are both [Rgb](#). To apply some transform to a color image, you initialize a color lookup once for all and use it at will in a transformation operation such as [EColorLookup::Transform](#).

Note. The offsets are specified as unquantized values.

EColorLookup.Calibrate

Sets a color transformation to recalibrate.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void Calibrate(  
    Euresys.Open_eVision_2_10.EC24 Color0,  
    float x0,  
    float y0,  
    float z0,  
    Euresys.Open_eVision_2_10.EC24 Color1,  
    float x1,  
    float y1,  
    float z1,  
    Euresys.Open_eVision_2_10.EC24 Color2,  
    float x2,  
    float y2,  
    float z2  
)
```

```
void Calibrate(  
    Euresys.Open_eVision_2_10.EC24 Color0,  
    float x0,  
    float y0,  
    float z0,  
    Euresys.Open_eVision_2_10.EC24 Color1,  
    float x1,  
    float y1,  
    float z1,  
    Euresys.Open_eVision_2_10.EC24 Color2,  
    float x2,  
    float y2,  
    float z2,  
    Euresys.Open_eVision_2_10.EC24 Color3,  
    float x3,  
    float y3,  
    float z3  
)
```

```
void Calibrate(  
    Euresys.Open_eVision_2_10.EC24 color,  
    float x,  
    float y,  
    float z  
)
```

Parameters

Color0

Measured quantized values of a pixel of color 0.

x_0

CIE XYZ tri-stimulus unquantized values corresponding to color 0.

y_0

CIE XYZ tri-stimulus unquantized values corresponding to color 0.

z_0

CIE XYZ tri-stimulus unquantized values corresponding to color 0.

Color1

Measured quantized values of a pixel of color 1.

x_1

CIE XYZ tri-stimulus unquantized values corresponding to color 1.

y_1

CIE XYZ tri-stimulus unquantized values corresponding to color 1.

z_1

CIE XYZ tri-stimulus unquantized values corresponding to color 1.

Color2

Measured quantized values of a pixel of color 2.

x_2

CIE XYZ tri-stimulus unquantized values corresponding to color 2.

y_2

CIE XYZ tri-stimulus unquantized values corresponding to color 2.

z_2

CIE XYZ tri-stimulus unquantized values corresponding to color 2.

Color3

Measured quantized values of a pixel of color 3.

x_3

CIE XYZ tri-stimulus unquantized values corresponding to color 3.

y_3

CIE XYZ tri-stimulus unquantized values corresponding to color 3.

z_3

CIE XYZ tri-stimulus unquantized values corresponding to color 3.

color

-

x

-

y

-

z

Remarks

The first prototype uses 3 reference colors. The second uses 4 reference colors. To apply some transform to a color image, you initialize a color lookup once for all and use it at will in a transformation operation such as [EColorLookup::Transform](#).

EColorLookup.ColorSystemIn

Input color system.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EColorSystem ColorSystemIn  
    { get; }
```

Remarks

The **EColorLookup** objects keep track of the color system transformation, for consistency. When applying a transformation, the source image color system (usually [Rgb](#)) must match the *input color system*; the destination image will be automatically be typed with the *output color system*. In case of a mismatch, an error message is issued. These two values are set by the lookup table initialization functions. An uninitialized lookup table has both color systems set to [NoColor](#).

EColorLookup.ColorSystemOut

Output color system.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EColorSystem ColorSystemOut  
{ get; }
```

Remarks

The **EColorLookup** objects keep track of the color system transformation, for consistency. When applying a transformation, the source image color system (usually [Rgb](#)) must match the *input color system*; the destination image will be automatically be typed with the *output color system*. In case of a mismatch, an error message is issued. These two values are set by the lookup table initialization functions. An uninitialized lookup table has both color systems set to [NoColor](#).

EColorLookup.ConvertFromRgb

Sets a color transformation from the [Rgb](#) representation to another system, as defined by [EColorSystem](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void ConvertFromRgb(  
    Euresys.Open_eVision_2_10.EColorSystem colorSystem  
)
```

Parameters

colorSystem

Color system, as defined by [EColorSystem](#).

Remarks

The input and output color systems are respectively [Rgb](#) and **colorSystem**. To apply some transform to a color image, you initialize a color lookup once for all and use it at will in a transformation operation such as [EColorLookup::Transform](#).

EColorLookup.ConvertToRgb

Sets a color transformation from any color system, as defined by [EColorSystem](#), to the [Rgb](#) representation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ConvertToRgb(
    Euresys.Open_eVision_2_10.EColorSystem colorSystem
)
```

Parameters

colorSystem

Color system, as defined by [EColorSystem](#).

Remarks

The input and output color systems are respectively **colorSystem** and [Rgb](#). To apply some transform to a color image, you initialize a color lookup once for all and use it at will in a transformation operation such as [EColorLookup::Transform](#).

EColorLookup.EColorLookup

Constructs a color lookup table.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EColorLookup(
    Euresys.Open_eVision_2_10.EColorLookup other
)
```

```
void EColorLookup(  
    )
```

Parameters

other

-

EColorLookup.IndexBits

Number of bits used for indexing the lookup table.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
uint IndexBits  
  
    { get; set; }
```

Remarks

Before filling in a lookup table, it is necessary to decide how many table entries it requires. The [EColorLookup::IndexBits](#) property indicates how many (high-order) bits of the input components are used. The relation between [EColorLookup::IndexBits](#), the number of table entries and the corresponding table size are given below: The larger the number of entries, the more accuracy is obtained. After [EColorLookup::IndexBits](#) has been changed, the lookup table needs to be recomputed.

Note. Be aware that each time a color lookup table is filled, all the entries are recomputed. When [EColorLookup::IndexBits](#) equals **6**, this may take a very long time. Such large lookup tables should be computed once only. Different combinations of [EColorLookup::IndexBits](#) and **Interpolation** provide a trade-off between accuracy and speed for the table pre-computation and table use.

EColorLookup.Interpolation

Interpolation mode.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool Interpolation
    { get; set; }
```

Remarks

When applying a lookup table to transform pixel values, tri-linear interpolation can be used: * when interpolation is not used, the table is looked up at the entry closest to the pixel value. This gives an accuracy equal to the value of the **IndexBits** property. On the other hand, table lookup is very fast; * when interpolation is used, the table is looked up at eight neighboring entries and an adequate average is computed. This gives full accuracy (8 bits) if the transformation is smooth enough. On the other hand, table lookup is slower.

Note. The interpolation mode may be modified at any time without the need to reinitialize the lookup table.

EColorLookup.Transform

Transforms a quantized color image/pixel to another quantized color image/pixel, using the previously initialized color lookup table.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Transform(
    Euresys.Open_eVision_2_10.EC24 sourceImageColor,
    out Euresys.Open_eVision_2_10.EC24 destinationImageColor
)

void Transform(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage
)
```

Parameters

sourceImageColor

Input color image.

destinationImageColor

Output color image.

sourceImage

Input color image.

destinationImage

Output color image.

EColorLookup.WhiteBalance

Initializes a color lookup table that can be used for color adjustment, including a global gain, gamma pre-compensation [cancellation] and white balancing.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void WhiteBalance(
    float gain,
    float gamma,
    float balanceRed,
    float balanceGreen,
    float balanceBlue
)
```

Parameters

gain

Global gain to be applied to all three color components. By default, the image intensity remains unchanged.

gamma

Gamma exponent. Setting this parameter will cancel the gamma pre-compensation feature of the camera, or apply it. By default, the no gamma pre-compensation is assumed (linear response). The gamma exponent can be chosen among the predefined values

[EasyColor::CompensateNtscGamma](#)/[EasyColor::CompensatePalGamma](#)/[EasyColor::CompensateSmpteGamma](#) (pre-compensation) or [EasyColor::NtscGamma](#)/[EasyColor::PalGamma](#)/[EasyColor::SmpteGamma](#) (pre-compensation cancellation), or be user-defined.

balanceRed

Color values to be used for white balance. These parameters should be set to the measured average values of a white (or gray) pixels area, allowing the white balance to adjust pre-component gains appropriately. Use function [EasyImage::PixelAverage](#) to obtain them. By default, no white balancing is performed.

balanceGreen

Color values to be used for white balance. These parameters should be set to the measured average values of a white (or gray) pixels area, allowing the white balance to adjust pre-component gains appropriately. Use function [EasyImage::PixelAverage](#) to obtain them. By default, no white balancing is performed.

balanceBlue

Color values to be used for white balance. These parameters should be set to the measured average values of a white (or gray) pixels area, allowing the white balance to adjust pre-component gains appropriately. Use function [EasyImage::PixelAverage](#) to obtain them. By default, no white balancing is performed.

Remarks

To apply some transform to a color image, you initialize the color lookup once for all and use it at will with [EColorLookup::Transform](#) or [EasyColor::TransformBayer](#) operation.

4.50. EColorRangeThresholdSegmenter Class

Segments an image using a double threshold on a color image.

Remarks

This segmenter is applicable to [EROIC24](#) RGB color images. It produces coded images with two layers: The White layer (usually, with index 1) contains the unmasked pixels that belong to the cube of the RGB space that spans the low threshold point to the high threshold point; and the Black layer (usually, with index 0) contains the remaining unmasked pixels.

Base Class: [ETwoLayersImageSegmenter](#)

Namespace: [Euresys.Open_eVision_2_10.Segmenters](#)

Properties

[HighThreshold](#)

Value of the high threshold.

[LowThreshold](#)

Value of the low threshold.

EColorRangeThresholdSegmenter.HighThreshold

Value of the high threshold.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]
```

```
Euresys.Open_eVision_2_10.EC24 HighThreshold  
{ get; set; }
```

EColorRangeThresholdSegmenter.LowThreshold

Value of the low threshold.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]
```

```
Euresys.Open_eVision_2_10.EC24 LowThreshold  
{ get; set; }
```

4.51. EColorSingleThresholdSegmenter Class

Segments an image using a single threshold on a color image.

Remarks

This segmenter is applicable to [EROIC24](#) RGB color images. It produces coded images with two layers: The White layer (usually, with index 1) contains the unmasked pixels that belong to the cube of the RGB space defined by the threshold point and the white point (**255,255,255**); and the Black layer (usually, with index 0) contains the remaining unmasked pixels.

Base Class: [ETwoLayersImageSegmenter](#)

Namespace: Euresys.Open_eVision_2_10.Segmenters

Properties

Threshold

| E Value of the threshold.

C

olorSingleThresholdSegmenter.Threshold

Value of the threshold.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]
```

```
Euresys.Open_eVision_2_10.EC24 Threshold
```

```
{ get; set; }
```

4.52. EColorVector Class

Base Class: [EVector](#)

Namespace: Euresys.Open_eVision_2_10

Properties

RawDataPtr

M Pointer to the vector data.

e

Methods

AddElement

Appends (adds at the tail) an element to the vector.

EColorVector

-

GetElement

Returns the vector element at the given index.

operator[]

Gives access to the vector element at the given index.

operator=

Copies all the data from another EColorVector object into the current EColorVector object

SetElement

E Modifies the vector element at the given index by the given value.

C

EColorVector.AddElement

Appends (adds at the tail) an element to the vector.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddElement(
    Euresys.Open_eVision_2_10.EColor element
)
```

Parameters

element

The element to be added.

EColorVector.EColorVector

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EColorVector(
)
void EColorVector(
    uint un32MaxElements
)
void EColorVector(
    Euresys.Open_eVision_2_10.EColorVector other
)
```

Parameters

un32MaxElements

-

other

-

EColorVector.GetElement

Returns the vector element at the given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EColor GetElement(
    int index
)
```

Parameters

index

Index, between **0** and [EColorVector](#) (excluded) of the element to be accessed.

Remarks

If the given index is outside the bounds of the vector, the error code [Parameter1OutOfRange](#) is set.

EColorVector.operator[]

Gives access to the vector element at the given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
ref Euresys.Open_eVision_2_10.EColor operator[] (
    uint index
)
```

Parameters

index

Index, between **0** and [EColorVector](#) (excluded) of the element to be accessed.

EColorVector.operator=

Copies all the data from another EColorVector object into the current EColorVector object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EColorVector operator=(
    Euresys.Open_eVision_2_10.EColorVector other
)
```

Parameters

other

EColorVector object to be copied

EColorVector.RawDataPtr

Pointer to the vector data.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
IntPtr RawDataPtr
{ get; }
```

EColorVector.SetElement

Modifies the vector element at the given index by the given value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void SetElement(  
    int index,  
    Euresys.Open_eVision_2_10.EColor value  
)
```

Parameters

index

Index, between **0** and [EColorVector](#) (excluded), of the element to be modified.

value

The new value for the element.

Remarks

If the given index is outside the bounds of the vector, the error code [Parameter1OutOfRange](#) is set.

4.53. EConverter Class

Conversion functions between bit depth formats of various 3D classes.

Namespace: Euresys.Open_eVision_2_10.Easy3D

Methods

[Convert](#)

Converts [EDepthMap](#) or [EZMap](#) between various formats.

Converter.Convert

Converts [EDepthMap](#) or [EZMap](#) between various formats.

Namespace: Euresys.Open_eVision_2_10.Easy3D

[C#]

```
void Convert(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 DepthMapIn,  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 DepthMapOut,  
    Euresys.Open_eVision_2_10.Easy3D.EMapConversionMode ConversionMode  
)
```

```
void Convert(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 DepthMapIn,  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap32f DepthMapOut  
)
```

```
void Convert(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 DepthMapIn,  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 DepthMapOut,  
    Euresys.Open_eVision_2_10.Easy3D.EMapConversionMode ConversionMode  
)
```

```
void Convert(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 DepthMapIn,  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap32f DepthMapOut  
)
```

```
void Convert(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap32f DepthMapIn,  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 DepthMapOut  
)
```

```
void Convert(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap32f DepthMapIn,  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 DepthMapOut  
)
```

```
void Convert(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 ZMapIn,  
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 ZMapOut,  
    Euresys.Open_eVision_2_10.Easy3D.EMapConversionMode ConversionMode  
)
```

```
void Convert(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 ZMapIn,  
    Euresys.Open_eVision_2_10.Easy3D.EZMap32f ZMapOut  
)
```

```

void Convert(
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 ZMapIn,
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 ZMapOut,
    Euresys.Open_eVision_2_10.Easy3D.EMapConversionMode ConversionMode
)

void Convert(
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 ZMapIn,
    Euresys.Open_eVision_2_10.Easy3D.EZMap32f ZMapOut
)

void Convert(
    Euresys.Open_eVision_2_10.Easy3D.EZMap32f ZMapIn,
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 ZMapOut
)

void Convert(
    Euresys.Open_eVision_2_10.Easy3D.EZMap32f MapIn,
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 MapOut
)

```

Parameters

DepthMapIn

The input DepthMap (8, 16 or 32 bits)

DepthMapOut

The output DepthMap (8, 16 or 32 bits)

ConversionMode

The conversion mode from [EMapConversionMode](#) defines how to transform the pixel value from a format (8, 16 or 32 bits) to another.

[MaxDynamic](#) maximizes the used range.

[Shift](#) converts by bit shifting.

By default, the mode [MaxDynamic](#) is selected.

ZMapIn

The input ZMap (8, 16 or 32 bits)

ZMapOut

The output ZMap (8, 16 or 32 bits)

MapIn

-

MapOut

-

Remarks

Conversion from or to 32bits only supports [MaxDynamic](#) mode. The undefined values are converted to the new map undefined value. For 8 and 16 bits images, the minimum defined value is 1, so the conversion of 32 bits images to 8 or 16 bits images adapts the dynamic range between 1 and the maximum value. For conversion to 32 bits float image, the used dynamic range is between 0 and FLOATMAX.

4.54. EDecimator Class

Decimation of a point cloud/Zmap/Depthmap.

Derived Class(es): [ERandomDecimator](#)

Namespace: Euresys.Open_eVision_2_10.Easy3D

Methods

[Decimate](#)

Decimates a [EPointCloud](#).

[EDecimator](#)

Creates an [EDecimator](#) object.

[Load](#)

Loads the decimator configuration. The given [ESerializer](#) must have been created for reading.

[Save](#)

Saves the decimator configuration. The given [ESerializer](#) must have been created for writing.

D

ecimator.Decimate

Decimates a [EPointCloud](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Decimate (
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud cloudIn,
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud cloudOut
)
```

Parameters

cloudIn

The input point cloud.

cloudOut

The output point cloud.

EDecimator.EDecimator

Creates an [EDecimator](#) object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void EDecimator (
)
void EDecimator (
    Euresys.Open_eVision_2_10.Easy3D.EDecimator other
)
```

Parameters

other

-

EDecimator.Load

Loads the decimator configuration. The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Load(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

-

EDecimator.Save

Saves the decimator configuration. The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

-

4.55. EDepthMap Class

Represents a generic DepthMap type interface.

Derived Class(es): [EDepthMap8](#) [EDepthMap16](#) [EDepthMap32f](#)

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

AxisSystemType

Manage the axis coordinate system.

Height

Access depth map Height.

RowPitch

Returns the buffer row pitch.

Type

Returns the image type.

Width

Access depth map Width.

ZResolution

M Access the Z Resolution (depth units per grey scale value).

e

Methods

AddMetadata

Adds a metadata key (name) and value. Overwrites if exists.

Clear	Clears the depth map: replaces all pixels with undefined value
ClearMetadata	Deletes all metadata.
ConvertCoordinatesMapToPixel	Converts 3D Map coordinates to Pixel coordinates.
ConvertCoordinatesPixelToMap	Converts Pixel coordinates to 3D Map coordinates.
DeleteMetadata	Deletes an existing metadata. Throws an exception if it does not exist.
Draw	Draws a DepthMap in a device context.
DrawImage	Displays the internal image buffer
GetBufferPtr	Retrieves the pointer of the pixel buffer.
GetCheckedBufferPtr	Retrieves the pointer of the pixel buffer.
GetMetadata	Returns string value of the given metadata. Throws an exception if it does not exist.
GetZValue	Gets the Z value of a pixel.

IsVoid	Tests if the EDepthMap object has a size of zero.
Load	Restores the EDepthMap stored in the given Open eVision file.
LoadImage	Restores the EDepthMap image stored in the given image file.
LoadImageAndMetadata	Loads the image and the metadata from a file in JSON format.
LoadMetadata	Loads the metadata from a file in JSON format.
ModifyMetadata	Changes the value of an existing metadata. Throws an exception if the metadata does not exist.
Save	Saves the EDepthMap object to the given Open eVision file.
SaveImage	Saves the EDepthMap image to the given image file.
SaveImageAndMetadata	Saves the image and the metadata to a JSON format file.
SaveJpeg	Saves the EDepthMap object to the given image file, in JPEG format.
SaveJpeg2K	Saves the EDepthMap object to the given imagefile, in JPEG 2000 format.

SaveMetadata

Saves the metadata to a file in JSON format

Serialize

Serializes the object with all its attributes.

SerializeImage

Serializes the image associated to [EDepthMap](#).

SetBufferPtr

Sets the pointer to an externally allocated buffer.

SetSize

⌈ Sets the width and height of a DepthMap.

D

epthMap.AddMetadata

Adds a metadata key (name) and value. Overwrites if exists.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
void AddMetadata (  
    string Key,  
    string value  
)
```

Parameters

Key

The name of the metadata. Names are unique.

value

The value for the given metadata.

EDepthMap.AxisSystemType

Manage the axis coordinate system.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
abstract Euresys.Open_eVision_2_10.Easy3D.EAxisSystemType AxisSystemType  
    { get; set; }
```

EDepthMap.Clear

Clears the depth map: replaces all pixels with undefined value

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void Clear(  
    )
```

EDepthMap.ClearMetadata

Deletes all metadata.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ClearMetadata (
)
```

EDepthMap.ConvertCoordinatesMapToPixel

Converts 3D Map coordinates to Pixel coordinates.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool ConvertCoordinatesMapToPixel (
    float x3D,
    float y3D,
    ref int xBuffer,
    ref int yBuffer
)
```

Parameters

x3D

The Map X coordinate.

y3D

The Map Y coordinate.

xBuffer

The returned Pixel X coordinate.

yBuffer

The returned Pixel Y coordinate.

EDepthMap.ConvertCoordinatesPixelToMap

Converts Pixel coordinates to 3D Map coordinates.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ConvertCoordinatesPixelToMap (
    int xBuffer,
    int yBuffer,
    ref float x3D,
    ref float y3D
)
```

Parameters

xBuffer

The pixel X coordinate.

yBuffer

The pixel Y coordinate.

x3D

The returned Map X coordinate.

y3D

The returned Map Y coordinate.

EDepthMap.DeleteMetadata

Deletes an existing metadata.
Throws an exception if it does not exist.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void DeleteMetadata (
    string Key
)
```

Parameters

Key

The name of an existing metadata.

EDepthMap.Draw

Draws a DepthMap in a device context.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Draw (
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void Draw (
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```



```
void Draw(  
    IntPtr graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

Parameters

graphicContext

Handle to the device context of the destination window.

zoomX

Magnification factor for zooming in or out in the horizontal direction. By default, the depthmap is displayed in 1:1 scale.

zoomY

Magnification factor for zooming in or out in the vertical direction. Setting a **0** value (which is the default) will result in isotropic scaling (i.e. equal horizontal and vertical factors).

panX

Pan offset (in pixels) in the horizontal direction. By default, no panning is applied.

panY

Pan offset (in pixels) in the vertical direction. By default, no panning is applied.

colorUndefinedPixel

An optional parameter to choose the drawing color of undefined pixels.

c24Vector

When supplied, this parameter allows using a LUT that maps from Depth to C24 when drawing (false colors).

bw8Vector

When supplied, this parameter allows using a LUT that maps from Depth to BW8 when drawing.

Remarks

A DepthMap can be drawn (its pixels rendered) using a device context.

The horizontal and vertical zooming factors can be different but must be in the **1/16..16** range.

(MFC users can use the **CDC::GetSafeHdc()** method to obtain a suitable device context handle from a **CDC** instance.)

EDepthMap.DrawImage

Displays the internal image buffer

Namespace: Euresys.Open_eVision_2_10.Easy3D

[C#]

```
void DrawImage(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void DrawImage(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void DrawImage(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void DrawImage(  
    IntPtr graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```

void DrawImage(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

void DrawImage(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

```

Parameters

graphicContext

Handle to the device context of the destination window.

zoomX

Magnification factor for zooming in or out in the horizontal direction. By default, the image is displayed in 1:1 scale.

zoomY

Magnification factor for zooming in or out in the vertical direction. Setting a **0** value (which is the default) will result in isotropic scaling (i.e. equal horizontal and vertical factors).

panX

Pan offset (in pixels) in the horizontal direction. By default, no panning is applied.

panY

Pan offset (in pixels) in the vertical direction. By default, no panning is applied.

colorUndefinedPixel

An optional parameter to choose the drawing color of undefined pixels.

c24Vector

When supplied, this parameter allows using a LUT that maps from Depth to C24 when drawing (false colors).

bw8Vector

When supplied, this parameter allows using a LUT that maps from Depth to BW8 when drawing.

Remarks

An image can be drawn (its pixels rendered) using a device context.

The horizontal and vertical zooming factors can be different but must be in the **1/16..16** range.

(MFC users can use the **CDC::GetSafeHdc()** method to obtain a suitable device context handle from a **CDC** instance.)

EDepthMap.GetBufferPtr

Retrieves the pointer of the pixel buffer.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
IntPtr GetBufferPtr(
)
IntPtr GetBufferPtr(
    int x,
    int y
)
IntPtr GetBufferPtr(
)
IntPtr GetBufferPtr(
    int x,
    int y
)
```

Parameters

x
Column of the pixel of which we want the address.

y
Row of the pixel of which we want the address.

Remarks

This function does not check the value of the parameters.
Use carefully.

EDepthMap.GetCheckedBufferPtr

Retrieves the pointer of the pixel buffer.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
IntPtr GetCheckedBufferPtr (
    int x,
    int y
)

IntPtr GetCheckedBufferPtr (
    int x,
    int y
)
```

Parameters

- x*
Column of the pixel of which we want the address.
- y*
Row of the pixel of which we want the address.

Remarks

This function checks the value of the parameters.

EDepthMap.GetMetadata

Returns string value of the given metadata.
Throws an exception if it does not exist.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
string GetMetadata (
    string Key
)
```

Parameters

Key

The name of an existing metadata.

EDepthMap.GetZValue

Gets the Z value of a pixel.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
float GetZValue (
    int x,
    int y
)
```

Parameters

x

Column of the pixel.

y

Row of the pixel.

EDepthMap.Height

Access depth map Height.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
abstract int Height
{ get; set; }
```

EDepthMap.IsVoid

Tests if the [EDepthMap](#) object has a size of zero.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool IsVoid(
)
```

Remarks

Returns **TRUE** if the depthmap size is zero.

EDepthMap.Load

Restores the [EDepthMap](#) stored in the given Open eVision file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Load(
    string path
)
```


Parameters

path

Full path of the file.

Remarks

When loading, the depth map is resized if needed.

This function restores the depth map attributes.

EDepthMap.LoadImage

Restores the [EDepthMap](#) image stored in the given image file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void LoadImage(
    string path
)
```

Parameters

path

Full path to the file.

Remarks

When loading, the depth map is resized if need be.

This function does not restore the depth map attributes, only the image associated with the [EDepthMap](#) is updated.

EDepthMap.LoadImageAndMetadata

Loads the image and the metadata from a file in JSON format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void LoadImageAndMetadata (
    string pathImage,
    string pathMetadata
)
```

Parameters

pathImage

Full path to the image file.

pathMetadata

Full path to the metadata file.

EDepthMap.LoadMetadata

Loads the metadata from a file in JSON format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void LoadMetadata (
    string path
)
```

Parameters

path

Full path to the file.

EDepthMap.ModifyMetadata

Changes the value of an existing metadata.
Throws an exception if the metadata does not exist.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ModifyMetadata (
    string Key,
    string value
)
```

Parameters

Key

The name of an existing metadata.

value

The value for the given metadata.

EDepthMap.RowPitch

Returns the buffer row pitch.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
abstract int RowPitch
{ get; }
```

EDepthMap.Save

Saves the [EDepthMap](#) object to the given Open eVision file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Save(
    string path
)
```

Parameters

path

The full path to the destination file.

Remarks

This function saves the [EDepthMap](#) in a Open eVision file.
This function stores the depth map attributes.

EDepthMap.SaveImage

Saves the [EDepthMap](#) image to the given image file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SaveImage(
    string path,
    Euresys.Open_eVision_2_10.EImageFileType type
)
```

Parameters

path

The full path to the destination file.

type

File format, as defined by [EImageFileType](#).

If not specified, the file format is determined from the file extension.

Remarks

This function saves the image associated to [EDepthMap](#) in a standard image file and thus does not store depth map attributes.

EDepthMap.SaveImageAndMetadata

Saves the image and the metadata to a JSON format file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SaveImageAndMetadata (
    string pathImage,
    string pathMetadata,
    Euresys.Open_eVision_2_10.EImageFileType type
)
```

Parameters

pathImage

The full path to the destination image file.

pathMetadata

The full path to the destination metadata file.

type

File format, as defined by [EImageFileType](#).

If not specified, the file format is determined from the file extension.

EDepthMap.SaveJpeg

Saves the [EDepthMap](#) object to the given image file, in JPEG format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
void SaveJpeg(  
    string path,  
    int quality  
)
```

Parameters

path

The full path of the destination file.

quality

JPEG quality, between 0 and 100 (100 is best quality). The default value is **75**.

EDepthMap.SaveJpeg2K

Saves the [EDepthMap](#) object to the given imagefile, in JPEG 2000 format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void SaveJpeg2K(  
    string path,  
    int quality  
)
```

Parameters

path

The full path of the destination file.

quality

JPEG 2000 quality, between 1 and 512.

The default value is **16**.

EDepthMap.SaveMetadata

Saves the metadata to a file in JSON format

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SaveMetadata(
    string path
)
```

Parameters

path

The full path to the destination file.

EDepthMap.Serialize

Serializes the object with all its attributes.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Serialize(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is read from or written to.

EDepthMap.SerializeImage

Serializes the image associated to [EDepthMap](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SerializeImage(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is read from or written to.

EDepthMap.SetBufferPtr

Sets the pointer to an externally allocated buffer.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetBufferPtr(
    int width,
    int height,
    IntPtr imagePointer,
    int bitsPerRow
)
```

Parameters

width

The width of the supplied buffer, in pixels.

height

The height of the supplied buffer, in pixels.

imagePointer

The pointer (aligned on 4 bytes) to the buffer, which must be large enough to hold the image.

bitsPerRow

The total number of bits contained in a row, padding included. Using the value **0** (default) means that this size is computed from the buffer width and the pixel size plus a padding with the smallest possible value that leads to a multiple of 4 bytes (32 bits), which is the minimum padding accepted by [EDepthMap::SetBufferPtr](#).

EDepthMap.SetSize

Sets the width and height of a DepthMap.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetSize(
    int width,
    int height
)

void SetSize(
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap other
)
```

Parameters

width

The new requested DepthMap width.

height

The new requested DepthMap height.

other

The other DepthMap whose dimensions have to be used for the current object.

Remarks

Open eVision will allocate a new buffer (deallocate the old buffer) if the supplied width and height are different from the existing ones.

If an external buffer has been specified by means of **SetImagePtr**, it will be kept only if the size does not change.

Creating a new buffer and setting its size creates a 4-byte aligned buffer, by default.

EDepthMap.Type

Returns the image type.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
abstract Euresys.Open_eVision_2_10.EImageType Type  
    { get; }
```

EDepthMap.Width

Access depth map Width.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
abstract int Width  
    { get; set; }
```

EDepthMap.ZResolution

Access the Z Resolution (depth units per grey scale value).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
abstract float ZResolution
```

```
{ get; set; }
```

4.56. EDepthMap16 Class

Represents a [EDepthMap](#) with an 16-bit pixel internal representation.

Base Class: [EDepthMap](#)

Derived Class(es): [EGrabberDepthMap16](#)

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

[AxisSystemType](#)

Manage the axis coordinate system.

[Height](#)

Access depth map Height.

[RowPitch](#)

Returns the buffer row pitch.

[Type](#)

Returns the image type.

[UndefinedValue](#)

Returns the Undefined value. That value is used to mark pixels with no valid depth value.

[Width](#)

Access depth map Width.

ZResolution

M Access the Z Resolution (depth units per grey scale value).

e

thods

AddMetadata

Adds a metadata key (name) and value. Overwrites if exists.

AsEImage

Casts the [EDepthMap16](#) to an [EImageBW16](#) to use with the Open eVision 2D tools.

Clear

Clears the depth map: replaces all pixels with undefined value

ClearMetadata

Deletes all metadata.

ConvertCoordinatesMapToPixel

Converts 3D Map coordinates to Pixel coordinates.

ConvertCoordinatesPixelToMap

Converts Pixel coordinates to 3D Map coordinates.

CopyMetadataTo

Copies all metadatas to another [EDepthMap16](#).

DeleteMetadata

Deletes an existing metadata.
Throws an exception if it does not exist.

Draw

Draws a DepthMap in a device context.

DrawImage	Displays the internal image buffer
EDepthMap16	Creates a 16 bits EDepthMap .
FillUndefinedPixels	Fills undefined pixels, used to fill the "holes" in the depth map.
GetBufferPtr	Retrieves the pointer of the pixel buffer.
GetCheckedBufferPtr	Retrieves the pointer of the pixel buffer.
GetMetadata	Returns string value of the given metadata. Throws an exception if it does not exist.
GetPixel	Gets the value of a pixel.
GetZValue	Gets Z value of a pixel.
IsVoid	Tests if the EDepthMap16 object has a size of zero.
Load	Restores the EDepthMap16 stored in the given Open eVision file.
LoadImage	Restores the EDepthMap16 image stored in the given image file.
LoadImageAndMetadata	Loads the image and the metadata from a file in JSON format.

LoadMetadata

Loads the metadata from a file in JSON format.

ModifyMetadata

Changes the value of an existing metadata.
Throws an exception if the metadata does not exist.

operator=

Assignment operator.

Save

Saves the [EDepthMap16](#) object to the given Open eVision file.

SaveImage

Saves the [EDepthMap16](#) image to the given image file.

SaveImageAndMetadata

Saves the image and the metadata to a JSON format file.

SaveJpeg

Saves the [EDepthMap16](#) object to the given image file, in JPEG format.

SaveJpeg2K

Saves the [EDepthMap16](#) object to the given imagefile, in JPEG 2000 format.

SaveMetadata

Saves the metadata to a file in JSON format

Serialize

Serializes the object with all its attributes.

SerializeImage

Serializes the image associated to [EDepthMap16](#).

[SetBufferPtr](#)

Sets the pointer to an externally allocated buffer.

[SetPixel](#)

Sets the value of a pixel.

[SetSize](#)

ED Sets the width and height of a DepthMap.

D

EDepthMap16.AddMetadata

Adds a metadata key (name) and value. Overwrites if exists.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
void AddMetadata (  
    string Key,  
    string value  
)
```

Parameters

Key

The name of the metadata. Names are unique.

value

The value for the given metadata.

EDepthMap16.AsEImage

Casts the [EDepthMap16](#) to an [EImageBW16](#) to use with the Open eVision 2D tools.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.EImageBW16 AsEImage (  
)
```

EDepthMap16.AxisSystemType

Manage the axis coordinate system.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
override Euresys.Open_eVision_2_10.Easy3D.EAxisSystemType AxisSystemType  
{ get; set; }
```

EDepthMap16.Clear

Clears the depth map: replaces all pixels with undefined value

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void Clear (  
)
```


EDepthMap16.ClearMetadata

Deletes all metadata.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void ClearMetadata (  
)
```

EDepthMap16.ConvertCoordinatesMapToPixel

Converts 3D Map coordinates to Pixel coordinates.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
bool ConvertCoordinatesMapToPixel (  
    float x3D,  
    float y3D,  
    ref int xBuffer,  
    ref int yBuffer  
)
```

Parameters

x3D

The Map X coordinate.

y3D

The Map Y coordinate.

xBuffer

The returned Pixel X coordinate.

yBuffer

The returned Pixel Y coordinate.

EDepthMap16.ConvertCoordinatesPixelToMap

Converts Pixel coordinates to 3D Map coordinates.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ConvertCoordinatesPixelToMap(
    int xBuffer,
    int yBuffer,
    ref float x3D,
    ref float y3D
)
```

Parameters

xBuffer

The pixel X coordinate.

yBuffer

The pixel Y coordinate.

x3D

The returned Map X coordinate.

y3D

The returned Map Y coordinate.

EDepthMap16.CopyMetadataTo

Copies all metadatas to another [EDepthMap16](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void CopyMetadataTo(
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 other
)
```

Parameters

other

The destination EDepthMap16.

EDepthMap16.DeleteMetadata

Deletes an existing metadata.
Throws an exception if it does not exist.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void DeleteMetadata(
    string Key
)
```

Parameters

Key

The name of an existing metadata.

EDepthMap16.Draw

Draws a DepthMap in a device context.

Namespace: Euresys.Open_eVision_2_10.Easy3D

[C#]

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

```

Parameters

graphicContext

Handle to the device context of the destination window.

zoomX

Magnification factor for zooming in or out in the horizontal direction. By default, the depthmap is displayed in 1:1 scale.

zoomY

Magnification factor for zooming in or out in the vertical direction. Setting a **0** value (which is the default) will result in isotropic scaling (i.e. equal horizontal and vertical factors).

panX

Pan offset (in pixels) in the horizontal direction. By default, no panning is applied.

panY

Pan offset (in pixels) in the vertical direction. By default, no panning is applied.

colorUndefinedPixel

An optional parameter to choose the drawing color of undefined pixels.

c24Vector

When supplied, this parameter allows using a LUT that maps from Depth to C24 when drawing (false colors).

bw8Vector

When supplied, this parameter allows using a LUT that maps from Depth to BW8 when drawing.

Remarks

A DepthMap can be drawn (its pixels rendered) using a device context.

The horizontal and vertical zooming factors can be different but must be in the **1/16..16** range.

(MFC users can use the **CDC::GetSafeHdc()** method to obtain a suitable device context handle from a **CDC** instance.)

EDepthMap16.DrawImage

Displays the internal image buffer

Namespace: Euresys.Open_eVision_2_10.Easy3D

[C#]

```
void DrawImage(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void DrawImage(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void DrawImage(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```

void DrawImage(
    IntPtr graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

void DrawImage(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

void DrawImage(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

```

Parameters

graphicContext

Handle to the device context of the destination window.

zoomX

Magnification factor for zooming in or out in the horizontal direction. By default, the image is displayed in 1:1 scale.

zoomY

Magnification factor for zooming in or out in the vertical direction. Setting a **0** value (which is the default) will result in isotropic scaling (i.e. equal horizontal and vertical factors).

panX

Pan offset (in pixels) in the horizontal direction. By default, no panning is applied.

panY

Pan offset (in pixels) in the vertical direction. By default, no panning is applied.

colorUndefinedPixel

An optional parameter to choose the drawing color of undefined pixels.

c24Vector

When supplied, this parameter allows using a LUT that maps from Depth to C24 when drawing (false colors).

bw8Vector

When supplied, this parameter allows using a LUT that maps from Depth to BW8 when drawing.

Remarks

An image can be drawn (its pixels rendered) using a device context.

The horizontal and vertical zooming factors can be different but must be in the **1/16..16** range.

(MFC users can use the **CDC::GetSafeHdc()** method to obtain a suitable device context handle from a **CDC** instance.)

EDepthMap16.EDepthMap16

Creates a 16 bits [EDepthMap](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void EDepthMap16 (
)
void EDepthMap16 (
    int width,
    int height
)
void EDepthMap16 (
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 other
)
```

Parameters

width

The width of the new depth map.

height

The height of the new depth map.

other

Another depth map.

EDepthMap16.FillUndefinedPixels

Fills undefined pixels, used to fill the "holes" in the depth map.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void FillUndefinedPixels (
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 outMap,
    Euresys.Open_eVision_2_10.Easy3D.EFillUndefinedPixelsDirection direction,
    Euresys.Open_eVision_2_10.Easy3D.EFillUndefinedPixelsMethod method
)
```

Parameters

outMap

The destination depth map.

direction

Direction in which the undefined pixels are filled in a depthmap from [EFillUndefinedPixelsDirection](#).

method

Which values used to fill the undefined pixels in a depthmap from [EFillUndefinedPixelsMethod](#).

EDepthMap16.GetBufferPtr

Retrieves the pointer of the pixel buffer.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
IntPtr GetBufferPtr (
)
```

```
IntPtr GetBufferPtr(  
    int x,  
    int y  
)  
  
IntPtr GetBufferPtr(  
)  
  
IntPtr GetBufferPtr(  
    int x,  
    int y  
)
```

Parameters

x

Column of the pixel of which we want the address.

y

Row of the pixel of which we want the address.

Remarks

This function does not check the value of the parameters.
Use carefully.

EDepthMap16.GetCheckedBufferPtr

Retrieves the pointer of the pixel buffer.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
IntPtr GetCheckedBufferPtr(  
    int x,  
    int y  
)
```

```
IntPtr GetCheckedBufferPtr(  
    int x,  
    int y  
)
```

Parameters

- x*
Column of the pixel of which we want the address.
- y*
Row of the pixel of which we want the address.

Remarks

This function checks the value of the parameters.

EDepthMap16.GetMetadata

Returns string value of the given metadata.
Throws an exception if it does not exist.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
string GetMetadata(  
    string Key  
)
```

Parameters

- Key*
The name of an existing metadata.

EDepthMap16.GetPixel

Gets the value of a pixel.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.EDepth16 GetPixel(
    int x,
    int y
)
```

Parameters

- x*
Column of the pixel.
- y*
Row of the pixel.

EDepthMap16.GetZValue

Gets Z value of a pixel.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
float GetZValue(
    int x,
    int y
)
```

Parameters

- x*

Column of the pixel.

y

Row of the pixel.

EDepthMap16.Height

Access depth map Height.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override int Height
{ get; set; }
```

EDepthMap16.IsVoid

Tests if the [EDepthMap16](#) object has a size of zero.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool IsVoid(
)
```

Remarks

Returns **TRUE** if the depthmap size is zero.

EDepthMap16.Load

Restores the [EDepthMap16](#) stored in the given Open eVision file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Load(
    string path
)
```

Parameters

path

Full path of the file.

Remarks

When loading, the depth map is resized if needed.

This function restores the depth map attributes.

EDepthMap16.LoadImage

Restores the [EDepthMap16](#) image stored in the given image file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void LoadImage(
    string path
)
```

Parameters

path

Full path to the file.

Remarks

When loading, the depth map is resized if need be.

This function does not restore the depth map attributes, only the image associated with the [EDepthMap16](#) is updated.

EDepthMap16.LoadImageAndMetadata

Loads the image and the metadata from a file in JSON format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
void LoadImageAndMetadata (  
    string pathImage,  
    string pathMetadata  
)
```

Parameters

pathImage

Full path to the image file.

pathMetadata

Full path to the metadata file.

EDepthMap16.LoadMetadata

Loads the metadata from a file in JSON format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```



```
void LoadMetadata(  
    string path  
)
```

Parameters

path

Full path to the file.

EDepthMap16.ModifyMetadata

Changes the value of an existing metadata.
Throws an exception if the metadata does not exist.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
void ModifyMetadata(  
    string Key,  
    string value  
)
```

Parameters

Key

The name of an existing metadata.

value

The value for the given metadata.

EDepthMap16.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 operator=(
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 other
)
```

Parameters

other

The source [EDepthMap16](#).

EDepthMap16.RowPitch

Returns the buffer row pitch.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override int RowPitch
{ get; }
```

EDepthMap16.Save

Saves the [EDepthMap16](#) object to the given Open eVision file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
void Save(  
    string path  
)
```

Parameters

path

The full path to the destination file.

Remarks

This function saves the [EDepthMap16](#) in a Open eVision file.
This function stores the depth map attributes.

EDepthMap16.SaveImage

Saves the [EDepthMap16](#) image to the given image file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
void SaveImage(  
    string path,  
    Euresys.Open_eVision_2_10.EImageFileType type  
)
```

Parameters

path

The full path to the destination file.

type

File format, as defined by [EImageFileType](#).

If not specified, the file format is determined from the file extension.

Remarks

This function saves the image associated to [EDepthMap16](#) in a standard image file and thus does not store depth map attributes.

EDepthMap16.SaveImageAndMetadata

Saves the image and the metadata to a JSON format file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SaveImageAndMetadata (
    string pathImage,
    string pathMetadata,
    Euresys.Open_eVision_2_10.EImageFileType type
)
```

Parameters

pathImage

The full path to the destination image file.

pathMetadata

The full path to the destination metadata file.

type

File format, as defined by [EImageFileType](#).

If not specified, the file format is determined from the file extension.

EDepthMap16.SaveJpeg

Saves the [EDepthMap16](#) object to the given image file, in JPEG format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
void SaveJpeg(  
    string path,  
    int quality  
)
```

Parameters

path

The full path of the destination file.

quality

JPEG quality, between 0 and 100 (100 is best quality). The default value is **75**.

EDepthMap16.SaveJpeg2K

Saves the [EDepthMap16](#) object to the given imagefile, in JPEG 2000 format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void SaveJpeg2K(  
    string path,  
    int quality  
)
```

Parameters

path

The full path of the destination file.

quality

JPEG 2000 quality, between 1 and 512.

The default value is **16**.

EDepthMap16.SaveMetadata

Saves the metadata to a file in JSON format

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SaveMetadata(
    string path
)
```

Parameters

path

The full path to the destination file.

EDepthMap16.Serialize

Serializes the object with all its attributes.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Serialize(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is read from or written to.

EDepthMap16.SerializeImage

Serializes the image associated to [EDepthMap16](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SerializeImage(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is read from or written to.

EDepthMap16.SetBufferPtr

Sets the pointer to an externally allocated buffer.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetBufferPtr(
    int width,
    int height,
    IntPtr imagePointer,
    int bitsPerRow
)
```

Parameters

width

The width of the supplied buffer, in pixels.

height

The height of the supplied buffer, in pixels.

imagePointer

The pointer (aligned on 4 bytes) to the buffer, which must be large enough to hold the image.

bitsPerRow

The total number of bits contained in a row, padding included. Using the value **0** (default) means that this size is computed from the buffer width and the pixel size plus a padding with the smallest possible value that leads to a multiple of 4 bytes (32 bits), which is the minimum padding accepted by [EDepthMap16::SetBufferPtr](#).

EDepthMap16.SetPixel

Sets the value of a pixel.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetPixel(
    Euresys.Open_eVision_2_10.EDepth16 value,
    int x,
    int y
)
```

Parameters

value

Value of the pixel.

x

Column of the pixel.

y

Row of the pixel.

EDepthMap16.SetSize

Sets the width and height of a DepthMap.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetSize(
    int width,
    int height
)
void SetSize(
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap other
)
```

Parameters

width

The new requested DepthMap width.

height

The new requested DepthMap height.

other

The other DepthMap whose dimensions have to be used for the current object.

Remarks

Open eVision will allocate a new buffer (deallocate the old buffer) if the supplied width and height are different from the existing ones.

If an external buffer has been specified by means of **SetImagePtr**, it will be kept only if the size does not change.

Creating a new buffer and setting its size creates a 4-byte aligned buffer, by default.

EDepthMap16.Type

Returns the image type.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override Euresys.Open_eVision_2_10.EImageType Type
```

```
{ get; }
```

EDepthMap16.UndefinedValue

Returns the Undefined value. That value is used to mark pixels with no valid depth value.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
Euresys.Open_eVision_2_10.EDepth16 UndefinedValue
```

```
{ get; }
```

EDepthMap16.Width

Access depth map Width.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
override int Width
```

```
{ get; set; }
```

EDepthMap16.ZResolution

Access the Z Resolution (depth units per grey scale value).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override float ZResolution
{ get; set; }
```

4.57. EDepthMap32f Class

Represents a [EDepthMap](#) with an 32-bit pixel internal representation.

Base Class: [EDepthMap](#)

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

[AxisSystemType](#)

Manage the axis coordinate system.

[Height](#)

Access depth map Height.

[RowPitch](#)

Returns the buffer row pitch.

[Type](#)

Returns the image type.

[UndefinedValue](#)

Returns the Undefined value. That value is used to mark pixels with no valid depth value.

[Width](#)

Access depth map Width.

ZResolution

M Access the Z Resolution (depth units per grey scale value).

e

thods

AddMetadata

Adds a metadata key (name) and value. Overwrites if exists.

AsEImage

Casts the [EDepthMap32f](#) to an [EImageBW32f](#) to use with the Open eVision 2D tools.

Clear

Clears the depth map: replaces all pixels with undefined value

ClearMetadata

Deletes all metadata.

ConvertCoordinatesMapToPixel

Converts 3D Map coordinates to Pixel coordinates.

ConvertCoordinatesPixelToMap

Converts Pixel coordinates to 3D Map coordinates.

CopyMetadataTo

Copies all metadatas to another [EDepthMap32f](#).

DeleteMetadata

Deletes an existing metadata.
Throws an exception if it does not exist.

Draw

Draws a DepthMap in a device context.

DrawImage	Displays the internal image buffer
EDepthMap32f	Creates a 32 bits EDepthMap .
FillUndefinedPixels	Fills undefined pixels, used to fill the "holes" in the depth map.
GetBufferPtr	Retrieves the pointer of the pixel buffer.
GetCheckedBufferPtr	Retrieves the pointer of the pixel buffer.
GetMetadata	Returns string value of the given metadata. Throws an exception if it does not exist.
GetPixel	Gets the value of a pixel.
GetZValue	Gets Z value of a pixel.
IsVoid	Tests if the EDepthMap32f object has a size of zero.
Load	Restores the EDepthMap32f stored in the given Open eVision file.
LoadImage	Restores the EDepthMap32f image stored in the given image file.
LoadImageAndMetadata	Loads the image and the metadata from a file in JSON format.

LoadMetadata

Loads the metadata from a file in JSON format.

ModifyMetadata

Changes the value of an existing metadata.
Throws an exception if the metadata does not exist.

operator=

Assignment operator.

Save

Saves the [EDepthMap32f](#) object to the given Open eVision file.

SaveImage

Saves the [EDepthMap32f](#) image to the given image file.

SaveImageAndMetadata

Saves the image and the metadata to a JSON format file.

SaveJpeg

Saves the [EDepthMap32f](#) object to the given image file, in JPEG format.

SaveJpeg2K

Saves the [EDepthMap32f](#) object to the given imagefile, in JPEG 2000 format.

SaveMetadata

Saves the metadata to a file in JSON format

Serialize

Serializes the object with all its attributes.

SerializeImage

Serializes the image associated to [EDepthMap32f](#).

[SetBufferPtr](#)

Sets the pointer to an externally allocated buffer.

[SetPixel](#)

Sets the value of a pixel.

[SetSize](#)

E Sets the width and height of a DepthMap.

D

[EDepthMap32f.AddMetadata](#)

Adds a metadata key (name) and value. Overwrites if exists.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void AddMetadata (
    string Key,
    string value
)
```

Parameters

Key

The name of the metadata. Names are unique.

value

The value for the given metadata.

[EDepthMap32f.AsEImage](#)

Casts the [EDepthMap32f](#) to an [EImageBW32f](#) to use with the Open eVision 2D tools.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.EImageBW32f AsEImage (  
)
```

EDepthMap32f.AxisSystemType

Manage the axis coordinate system.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
override Euresys.Open_eVision_2_10.Easy3D.EAxisSystemType AxisSystemType  
{ get; set; }
```

EDepthMap32f.Clear

Clears the depth map: replaces all pixels with undefined value

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void Clear (  
)
```


EDepthMap32f.ClearMetadata

Deletes all metadata.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void ClearMetadata (  
)
```

EDepthMap32f.ConvertCoordinatesMapToPixel

Converts 3D Map coordinates to Pixel coordinates.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
bool ConvertCoordinatesMapToPixel (  
    float x3D,  
    float y3D,  
    ref int xBuffer,  
    ref int yBuffer  
)
```

Parameters

x3D

The Map X coordinate.

y3D

The Map Y coordinate.

xBuffer

The returned Pixel X coordinate.

yBuffer

The returned Pixel Y coordinate.

EDepthMap32f.ConvertCoordinatesPixelToMap

Converts Pixel coordinates to 3D Map coordinates.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ConvertCoordinatesPixelToMap(
    int xBuffer,
    int yBuffer,
    ref float x3D,
    ref float y3D
)
```

Parameters

xBuffer

The pixel X coordinate.

yBuffer

The pixel Y coordinate.

x3D

The returned Map X coordinate.

y3D

The returned Map Y coordinate.

EDepthMap32f.CopyMetadataTo

Copies all metadatas to another [EDepthMap32f](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void CopyMetadataTo(
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap32f other
)
```

Parameters

other

The destination EDepthMap32f.

EDepthMap32f.DeleteMetadata

Deletes an existing metadata.
Throws an exception if it does not exist.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void DeleteMetadata(
    string Key
)
```

Parameters

Key

The name of an existing metadata.

EDepthMap32f.Draw

Draws a DepthMap in a device context.

Namespace: Euresys.Open_eVision_2_10.Easy3D

[C#]

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

```

Parameters

graphicContext

Handle to the device context of the destination window.

zoomX

Magnification factor for zooming in or out in the horizontal direction. By default, the depthmap is displayed in 1:1 scale.

zoomY

Magnification factor for zooming in or out in the vertical direction. Setting a **0** value (which is the default) will result in isotropic scaling (i.e. equal horizontal and vertical factors).

panX

Pan offset (in pixels) in the horizontal direction. By default, no panning is applied.

panY

Pan offset (in pixels) in the vertical direction. By default, no panning is applied.

colorUndefinedPixel

An optional parameter to choose the drawing color of undefined pixels.

c24Vector

When supplied, this parameter allows using a LUT that maps from Depth to C24 when drawing (false colors).

bw8Vector

When supplied, this parameter allows using a LUT that maps from Depth to BW8 when drawing.

Remarks

A DepthMap can be drawn (its pixels rendered) using a device context.

The horizontal and vertical zooming factors can be different but must be in the **1/16..16** range.

(MFC users can use the **CDC::GetSafeHdc()** method to obtain a suitable device context handle from a **CDC** instance.)

EDepthMap32f.DrawImage

Displays the internal image buffer

Namespace: Euresys.Open_eVision_2_10.Easy3D

[C#]

```
void DrawImage(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void DrawImage(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void DrawImage(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```

void DrawImage(
    IntPtr graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

void DrawImage(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

void DrawImage(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

```

Parameters

graphicContext

Handle to the device context of the destination window.

zoomX

Magnification factor for zooming in or out in the horizontal direction. By default, the image is displayed in 1:1 scale.

zoomY

Magnification factor for zooming in or out in the vertical direction. Setting a **0** value (which is the default) will result in isotropic scaling (i.e. equal horizontal and vertical factors).

panX

Pan offset (in pixels) in the horizontal direction. By default, no panning is applied.

panY

Pan offset (in pixels) in the vertical direction. By default, no panning is applied.

colorUndefinedPixel

An optional parameter to choose the drawing color of undefined pixels.

c24Vector

When supplied, this parameter allows using a LUT that maps from Depth to C24 when drawing (false colors).

bw8Vector

When supplied, this parameter allows using a LUT that maps from Depth to BW8 when drawing.

Remarks

An image can be drawn (its pixels rendered) using a device context.

The horizontal and vertical zooming factors can be different but must be in the **1/16..16** range.

(MFC users can use the **CDC::GetSafeHdc()** method to obtain a suitable device context handle from a **CDC** instance.)

EDepthMap32f.EDepthMap32f

Creates a 32 bits [EDepthMap](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void EDepthMap32f(
)
void EDepthMap32f(
    int width,
    int height
)
void EDepthMap32f(
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap32f other
)
```

Parameters

width

The width of the new depth map.

height

The height of the new depth map.

other

Another depth map.

EDepthMap32f.FillUndefinedPixels

Fills undefined pixels, used to fill the "holes" in the depth map.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void FillUndefinedPixels (
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap32f outMap,
    Euresys.Open_eVision_2_10.Easy3D.EFillUndefinedPixelsDirection direction,
    Euresys.Open_eVision_2_10.Easy3D.EFillUndefinedPixelsMethod method
)
```

Parameters

outMap

The destination depth map

direction

Direction in which the undefined pixels are filled in a depthmap from [EFillUndefinedPixelsDirection](#).

method

Which values used to fill the undefined pixels in a depthmap from [EFillUndefinedPixelsMethod](#).

EDepthMap32f.GetBufferPtr

Retrieves the pointer of the pixel buffer.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
IntPtr GetBufferPtr (
)
```

```
IntPtr GetBufferPtr(  
    int x,  
    int y  
)  
  
IntPtr GetBufferPtr(  
)  
  
IntPtr GetBufferPtr(  
    int x,  
    int y  
)
```

Parameters

x

Column of the pixel of which we want the address.

y

Row of the pixel of which we want the address.

Remarks

This function does not check the value of the parameters.
Use carefully.

EDepthMap32f.GetCheckedBufferPtr

Retrieves the pointer of the pixel buffer.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
IntPtr GetCheckedBufferPtr(  
    int x,  
    int y  
)
```

```
IntPtr GetCheckedBufferPtr(  
    int x,  
    int y  
)
```

Parameters

x
Column of the pixel of which we want the address.

y
Row of the pixel of which we want the address.

Remarks

This function checks the value of the parameters.

EDepthMap32f.GetMetadata

Returns string value of the given metadata.
Throws an exception if it does not exist.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
string GetMetadata(  
    string Key  
)
```

Parameters

Key
The name of an existing metadata.

EDepthMap32f.GetPixel

Gets the value of a pixel.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.EDepth32f GetPixel(  
    int x,  
    int y  
)
```

Parameters

- x*
Column of the pixel.
- y*
Row of the pixel.

EDepthMap32f.GetZValue

Gets Z value of a pixel.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float GetZValue(  
    int x,  
    int y  
)
```

Parameters

- x*

Column of the pixel.

y

Row of the pixel.

EDepthMap32f.Height

Access depth map Height.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
override int Height  
  
    { get; set; }
```

EDepthMap32f.IsVoid

Tests if the [EDepthMap32f](#) object has a size of zero.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
bool IsVoid(  
    )
```

Remarks

Returns **TRUE** if the depthmap size is zero.

EDepthMap32f.Load

Restores the [EDepthMap32f](#) stored in the given Open eVision file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Load(
    string path
)
```

Parameters

path
Full path of the file.

Remarks

When loading, the depth map is resized if needed.
This function restores the depth map attributes.

EDepthMap32f.LoadImage

Restores the [EDepthMap32f](#) image stored in the given image file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void LoadImage(
    string path
)
```

Parameters

path

Full path to the file.

Remarks

When loading, the depth map is resized if need be.

This function does not restore the depth map attributes, only the image associated with the [EDepthMap32f](#) is updated.

EDepthMap32f.LoadImageAndMetadata

Loads the image and the metadata from a file in JSON format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
void LoadImageAndMetadata (  
    string pathImage,  
    string pathMetadata  
)
```

Parameters

pathImage

Full path to the image file.

pathMetadata

Full path to the metadata file.

EDepthMap32f.LoadMetadata

Loads the metadata from a file in JSON format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```



```
void LoadMetadata(  
    string path  
)
```

Parameters

path

Full path to the file.

EDepthMap32f.ModifyMetadata

Changes the value of an existing metadata.
Throws an exception if the metadata does not exist.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
void ModifyMetadata(  
    string Key,  
    string value  
)
```

Parameters

Key

The name of an existing metadata.

value

The value for the given metadata.

EDepthMap32f.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EDepthMap32f operator=(
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap32f other
)
```

Parameters

other

The source [EDepthMap32f](#).

EDepthMap32f.RowPitch

Returns the buffer row pitch.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override int RowPitch
{ get; }
```

EDepthMap32f.Save

Saves the [EDepthMap32f](#) object to the given Open eVision file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
void Save(  
    string path  
)
```

Parameters

path

The full path to the destination file.

Remarks

This function saves the [EDepthMap32f](#) in a Open eVision file.
This function stores the depth map attributes.

EDepthMap32f.SaveImage

Saves the [EDepthMap32f](#) image to the given image file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
void SaveImage(  
    string path,  
    Euresys.Open_eVision_2_10.EImageFileType type  
)
```

Parameters

path

The full path to the destination file.

type

File format, as defined by [EImageFileType](#).

If not specified, the file format is determined from the file extension.

Remarks

This function saves the image associated to [EDepthMap32f](#) in a standard image file and thus does not store depth map attributes.

EDepthMap32f.SaveImageAndMetadata

Saves the image and the metadata to a JSON format file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SaveImageAndMetadata (
    string pathImage,
    string pathMetadata,
    Euresys.Open_eVision_2_10.EImageFileType type
)
```

Parameters

pathImage

The full path to the destination image file.

pathMetadata

The full path to the destination metadata file.

type

File format, as defined by [EImageFileType](#).

If not specified, the file format is determined from the file extension.

EDepthMap32f.SaveJpeg

Saves the [EDepthMap32f](#) object to the given image file, in JPEG format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
void SaveJpeg(  
    string path,  
    int quality  
)
```

Parameters

path

The full path of the destination file.

quality

JPEG quality, between 0 and 100 (100 is best quality). The default value is **75**.

EDepthMap32f.SaveJpeg2K

Saves the [EDepthMap32f](#) object to the given imagefile, in JPEG 2000 format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void SaveJpeg2K(  
    string path,  
    int quality  
)
```

Parameters

path

The full path of the destination file.

quality

JPEG 2000 quality, between 1 and 512.

The default value is **16**.

EDepthMap32f.SaveMetadata

Saves the metadata to a file in JSON format

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SaveMetadata(
    string path
)
```

Parameters

path

The full path to the destination file.

EDepthMap32f.Serialize

Serializes the object with all its attributes.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Serialize(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is read from or written to.

EDepthMap32f.SerializeImage

Serializes the image associated to [EDepthMap32f](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SerializeImage(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is read from or written to.

EDepthMap32f.SetBufferPtr

Sets the pointer to an externally allocated buffer.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetBufferPtr(
    int width,
    int height,
    IntPtr imagePointer,
    int bitsPerRow
)
```

Parameters

width

The width of the supplied buffer, in pixels.

height

The height of the supplied buffer, in pixels.

imagePointer

The pointer (aligned on 4 bytes) to the buffer, which must be large enough to hold the image.

bitsPerRow

The total number of bits contained in a row, padding included. Using the value **0** (default) means that this size is computed from the buffer width and the pixel size plus a padding with the smallest possible value that leads to a multiple of 4 bytes (32 bits), which is the minimum padding accepted by [EDepthMap32f::SetBufferPtr](#).

EDepthMap32f.SetPixel

Sets the value of a pixel.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetPixel(
    Euresys.Open_eVision_2_10.EDepth32f value,
    int x,
    int y
)
```

Parameters

value

Value of the pixel.

x

Column of the pixel.

y

Row of the pixel.

EDepthMap32f.SetSize

Sets the width and height of a DepthMap.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetSize(
    int width,
    int height
)

void SetSize(
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap other
)
```

Parameters

width

The new requested DepthMap width.

height

The new requested DepthMap height.

other

The other DepthMap whose dimensions have to be used for the current object.

Remarks

Open eVision will allocate a new buffer (deallocate the old buffer) if the supplied width and height are different from the existing ones.

If an external buffer has been specified by means of **SetImagePtr**, it will be kept only if the size does not change.

Creating a new buffer and setting its size creates a 4-byte aligned buffer, by default.

EDepthMap32f.Type

Returns the image type.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override Euresys.Open_eVision_2_10.EImageType Type
```

```
{ get; }
```

EDepthMap32f.UndefinedValue

Returns the Undefined value. That value is used to mark pixels with no valid depth value.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.EDepth32f UndefinedValue  
  
{ get; }
```

EDepthMap32f.Width

Access depth map Width.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
override int Width  
  
{ get; set; }
```

EDepthMap32f.ZResolution

Access the Z Resolution (depth units per grey scale value).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override float ZResolution
{ get; set; }
```

4.58. EDepthMap8 Class

Represents a [EDepthMap](#) with an internal 8-bit pixel representation.

Base Class: [EDepthMap](#)

Derived Class(es): [EGrabberDepthMap8](#)

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

[AxisSystemType](#)

Manage the axis coordinate system.

[Height](#)

Access depth map Height.

[RowPitch](#)

Returns the buffer row pitch.

[Type](#)

Returns the image type.

[UndefinedValue](#)

Returns the Undefined value. That value is used to mark pixels with no valid depth value.

[Width](#)

Access depth map Width.

ZResolution

M Access the Z Resolution (depth units per grey scale value).

e

thods

AddMetadata

Adds a metadata key (name) and value. Overwrites if exists.

AsEImage

Casts the [EDepthMap8](#) to an [EImageBW8](#) to use with the Open eVision 2D tools.

Clear

Clears the depth map: replaces all pixels with undefined value

ClearMetadata

Deletes all metadata.

ConvertCoordinatesMapToPixel

Converts 3D Map coordinates to Pixel coordinates.

ConvertCoordinatesPixelToMap

Converts Pixel coordinates to 3D Map coordinates.

CopyMetadataTo

Copies all metadatas to another [EDepthMap8](#).

DeleteMetadata

Deletes an existing metadata.
Throws an exception if it does not exist.

Draw

Draws a DepthMap in a device context.

DrawImage	Displays the internal image buffer
EDepthMap8	Creates a 8 bits EDepthMap .
FillUndefinedPixels	Fills undefined pixels, used to fill the "holes" in the depth map.
GetBufferPtr	Retrieves the pointer of the pixel buffer.
GetCheckedBufferPtr	Retrieves the pointer of the pixel buffer.
GetMetadata	Returns string value of the given metadata. Throws an exception if it does not exist.
GetPixel	Gets the value of a pixel.
GetZValue	Gets Z value of a pixel.
IsVoid	Tests if the EDepthMap8 object has a size of zero.
Load	Restores the EDepthMap8 stored in the given Open eVision file.
LoadImage	Restores the EDepthMap8 image stored in the given image file.
LoadImageAndMetadata	Loads the image and the metadata from a file in JSON format.

LoadMetadata

Loads the metadata from a file in JSON format.

ModifyMetadata

Changes the value of an existing metadata.
Throws an exception if the metadata does not exist.

operator=

Assignment operator.

Save

Saves the [EDepthMap8](#) object to the given Open eVision file.

SaveImage

Saves the [EDepthMap8](#) image to the given image file.

SaveImageAndMetadata

Saves the image and the metadata to a JSON format file.

SaveJpeg

Saves the [EDepthMap8](#) object to the given image file, in JPEG format.

SaveJpeg2K

Saves the [EDepthMap8](#) object to the given imagefile, in JPEG 2000 format.

SaveMetadata

Saves the metadata to a file in JSON format

Serialize

Serializes the object with all its attributes.

SerializeImage

Serializes the image associated to [EDepthMap8](#).

[SetBufferPtr](#)

Sets the pointer to an externally allocated buffer.

[SetPixel](#)

Sets the value of a pixel.

[SetSize](#)

E Sets the width and height of a DepthMap.

D

epthMap8.AddMetadata

Adds a metadata key (name) and value. Overwrites if exists.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void AddMetadata (
    string Key,
    string value
)
```

Parameters

Key

The name of the metadata. Names are unique.

value

The value for the given metadata.

EDepthMap8.AsEImage

Casts the [EDepthMap8](#) to an [EImageBW8](#) to use with the Open eVision 2D tools.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.EImageBW8 AsEImage (  
)
```

EDepthMap8.AxisSystemType

Manage the axis coordinate system.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
override Euresys.Open_eVision_2_10.Easy3D.EAxisSystemType AxisSystemType  
{ get; set; }
```

EDepthMap8.Clear

Clears the depth map: replaces all pixels with undefined value

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void Clear (  
)
```


EDepthMap8.ClearMetadata

Deletes all metadata.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void ClearMetadata (  
)
```

EDepthMap8.ConvertCoordinatesMapToPixel

Converts 3D Map coordinates to Pixel coordinates.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
bool ConvertCoordinatesMapToPixel (  
    float x3D,  
    float y3D,  
    ref int xBuffer,  
    ref int yBuffer  
)
```

Parameters

x3D

The Map X coordinate.

y3D

The Map Y coordinate.

xBuffer

The returned Pixel X coordinate.

yBuffer

The returned Pixel Y coordinate.

EDepthMap8.ConvertCoordinatesPixelToMap

Converts Pixel coordinates to 3D Map coordinates.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ConvertCoordinatesPixelToMap(
    int xBuffer,
    int yBuffer,
    ref float x3D,
    ref float y3D
)
```

Parameters

xBuffer

The pixel X coordinate.

yBuffer

The pixel Y coordinate.

x3D

The returned Map X coordinate.

y3D

The returned Map Y coordinate.

EDepthMap8.CopyMetadataTo

Copies all metadatas to another [EDepthMap8](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void CopyMetadataTo(
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 other
)
```

Parameters

other

The destination EDepthMap8.

EDepthMap8.DeleteMetadata

Deletes an existing metadata.
Throws an exception if it does not exist.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void DeleteMetadata(
    string Key
)
```

Parameters

Key

The name of an existing metadata.

EDepthMap8.Draw

Draws a DepthMap in a device context.

Namespace: Euresys.Open_eVision_2_10.Easy3D

[C#]

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

```

Parameters

graphicContext

Handle to the device context of the destination window.

zoomX

Magnification factor for zooming in or out in the horizontal direction. By default, the depthmap is displayed in 1:1 scale.

zoomY

Magnification factor for zooming in or out in the vertical direction. Setting a **0** value (which is the default) will result in isotropic scaling (i.e. equal horizontal and vertical factors).

panX

Pan offset (in pixels) in the horizontal direction. By default, no panning is applied.

panY

Pan offset (in pixels) in the vertical direction. By default, no panning is applied.

colorUndefinedPixel

An optional parameter to choose the drawing color of undefined pixels.

c24Vector

When supplied, this parameter allows using a LUT that maps from Depth to C24 when drawing (false colors).

bw8Vector

When supplied, this parameter allows using a LUT that maps from Depth to BW8 when drawing.

Remarks

A DepthMap can be drawn (its pixels rendered) using a device context.

The horizontal and vertical zooming factors can be different but must be in the **1/16..16** range.

(MFC users can use the **CDC::GetSafeHdc()** method to obtain a suitable device context handle from a **CDC** instance.)

EDepthMap8.DrawImage

Displays the internal image buffer

Namespace: Euresys.Open_eVision_2_10.Easy3D

[C#]

```
void DrawImage(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void DrawImage(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void DrawImage(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```

void DrawImage(
    IntPtr graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

void DrawImage(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

void DrawImage(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

```

Parameters

graphicContext

Handle to the device context of the destination window.

zoomX

Magnification factor for zooming in or out in the horizontal direction. By default, the image is displayed in 1:1 scale.

zoomY

Magnification factor for zooming in or out in the vertical direction. Setting a **0** value (which is the default) will result in isotropic scaling (i.e. equal horizontal and vertical factors).

panX

Pan offset (in pixels) in the horizontal direction. By default, no panning is applied.

panY

Pan offset (in pixels) in the vertical direction. By default, no panning is applied.

colorUndefinedPixel

An optional parameter to choose the drawing color of undefined pixels.

c24Vector

When supplied, this parameter allows using a LUT that maps from Depth to C24 when drawing (false colors).

bw8Vector

When supplied, this parameter allows using a LUT that maps from Depth to BW8 when drawing.

Remarks

An image can be drawn (its pixels rendered) using a device context.

The horizontal and vertical zooming factors can be different but must be in the **1/16..16** range.

(MFC users can use the **CDC::GetSafeHdc()** method to obtain a suitable device context handle from a **CDC** instance.)

EDepthMap8.EDepthMap8

Creates a 8 bits [EDepthMap](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void EDepthMap8 (
)
void EDepthMap8 (
    int width,
    int height
)
void EDepthMap8 (
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 other
)
```

Parameters

width

The width of the new depth map.

height

The height of the new depth map.

other

Another depth map.

EDepthMap8.FillUndefinedPixels

Fills undefined pixels, used to fill the "holes" in the depth map.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void FillUndefinedPixels (
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 outMap,
    Euresys.Open_eVision_2_10.Easy3D.EFillUndefinedPixelsDirection direction,
    Euresys.Open_eVision_2_10.Easy3D.EFillUndefinedPixelsMethod method
)
```

Parameters

outMap

The destination depth map.

direction

Direction in which the undefined pixels are filled in a depthmap from [EFillUndefinedPixelsDirection](#).

method

Which values used to fill the undefined pixels in a depthmap from [EFillUndefinedPixelsMethod](#).

EDepthMap8.GetBufferPtr

Retrieves the pointer of the pixel buffer.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
IntPtr GetBufferPtr (
)
```

```
IntPtr GetBufferPtr(  
    int x,  
    int y  
)  
  
IntPtr GetBufferPtr(  
)  
  
IntPtr GetBufferPtr(  
    int x,  
    int y  
)
```

Parameters

x

Column of the pixel of which we want the address.

y

Row of the pixel of which we want the address.

Remarks

This function does not check the value of the parameters.
Use carefully.

EDepthMap8.GetCheckedBufferPtr

Retrieves the pointer of the pixel buffer.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
IntPtr GetCheckedBufferPtr(  
    int x,  
    int y  
)
```

```
IntPtr GetCheckedBufferPtr(  
    int x,  
    int y  
)
```

Parameters

- x*
Column of the pixel of which we want the address.
- y*
Row of the pixel of which we want the address.

Remarks

This function checks the value of the parameters.

EDepthMap8.GetMetadata

Returns string value of the given metadata.
Throws an exception if it does not exist.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
string GetMetadata(  
    string Key  
)
```

Parameters

- Key*
The name of an existing metadata.

EDepthMap8.GetPixel

Gets the value of a pixel.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.EDepth8 GetPixel(
    int x,
    int y
)
```

Parameters

- x*
Column of the pixel.
- y*
Row of the pixel.

EDepthMap8.GetZValue

Gets Z value of a pixel.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
float GetZValue(
    int x,
    int y
)
```

Parameters

- x*

Column of the pixel.

y

Row of the pixel.

EDepthMap8.Height

Access depth map Height.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override int Height
    { get; set; }
```

EDepthMap8.IsVoid

Tests if the [EDepthMap8](#) object has a size of zero.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool IsVoid(
)
```

Remarks

Returns **TRUE** if the depthmap size is zero.

EDepthMap8.Load

Restores the [EDepthMap8](#) stored in the given Open eVision file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Load(
    string path
)
```

Parameters

path
Full path of the file.

Remarks

When loading, the depth map is resized if needed.
This function restores the depth map attributes.

EDepthMap8.LoadImage

Restores the [EDepthMap8](#) image stored in the given image file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void LoadImage(
    string path
)
```

Parameters

path

Full path to the file.

Remarks

When loading, the depth map is resized if need be.

This function does not restore the depth map attributes, only the image associated with the [EDepthMap8](#) is updated.

EDepthMap8.LoadImageAndMetadata

Loads the image and the metadata from a file in JSON format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void LoadImageAndMetadata (
    string pathImage,
    string pathMetadata
)
```

Parameters

pathImage

Full path to the image file.

pathMetadata

Full path to the metadata file.

EDepthMap8.LoadMetadata

Loads the metadata from a file in JSON format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```



```
void LoadMetadata(  
    string path  
)
```

Parameters

path

Full path to the file.

EDepthMap8.ModifyMetadata

Changes the value of an existing metadata.
Throws an exception if the metadata does not exist.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
void ModifyMetadata(  
    string Key,  
    string value  
)
```

Parameters

Key

The name of an existing metadata.

value

The value for the given metadata.

EDepthMap8.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 operator=(
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 other
)
```

Parameters

other

The source [EDepthMap8](#).

EDepthMap8.RowPitch

Returns the buffer row pitch.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override int RowPitch
{ get; }
```

EDepthMap8.Save

Saves the [EDepthMap8](#) object to the given Open eVision file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
void Save(  
    string path  
)
```

Parameters

path

The full path to the destination file.

Remarks

This function saves the [EDepthMap8](#) in a Open eVision file.
This function stores the depth map attributes.

EDepthMap8.SaveImage

Saves the [EDepthMap8](#) image to the given image file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
void SaveImage(  
    string path,  
    Euresys.Open_eVision_2_10.EImageFileType type  
)
```

Parameters

path

The full path to the destination file.

type

File format, as defined by [EImageFileType](#).
If not specified, the file format is determined from the file extension.

Remarks

This function saves the image associated to [EDepthMap8](#) in a standard image file and thus does not store depth map attributes.

EDepthMap8.SaveImageAndMetadata

Saves the image and the metadata to a JSON format file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SaveImageAndMetadata (
    string pathImage,
    string pathMetadata,
    Euresys.Open_eVision_2_10.EImageFileType type
)
```

Parameters

pathImage

The full path to the destination image file.

pathMetadata

The full path to the destination metadata file.

type

File format, as defined by [EImageFileType](#).

If not specified, the file format is determined from the file extension.

EDepthMap8.SaveJpeg

Saves the [EDepthMap8](#) object to the given image file, in JPEG format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
void SaveJpeg(  
    string path,  
    int quality  
)
```

Parameters

path

The full path of the destination file.

quality

JPEG quality, between 0 and 100 (100 is best quality). The default value is **75**.

EDepthMap8.SaveJpeg2K

Saves the [EDepthMap8](#) object to the given imagefile, in JPEG 2000 format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void SaveJpeg2K(  
    string path,  
    int quality  
)
```

Parameters

path

The full path of the destination file.

quality

JPEG 2000 quality, between 1 and 512.

The default value is **16**.

EDepthMap8.SaveMetadata

Saves the metadata to a file in JSON format

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SaveMetadata(
    string path
)
```

Parameters

path

The full path to the destination file.

EDepthMap8.Serialize

Serializes the object with all its attributes.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Serialize(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is read from or written to.

EDepthMap8.SerializeImage

Serializes the image associated to [EDepthMap8](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SerializeImage(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is read from or written to.

EDepthMap8.SetBufferPtr

Sets the pointer to an externally allocated buffer.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetBufferPtr(
    int width,
    int height,
    IntPtr imagePointer,
    int bitsPerRow
)
```

Parameters

width

The width of the supplied buffer, in pixels.

height

The height of the supplied buffer, in pixels.

imagePointer

The pointer (aligned on 4 bytes) to the buffer, which must be large enough to hold the image.

bitsPerRow

The total number of bits contained in a row, padding included. Using the value **0** (default) means that this size is computed from the buffer width and the pixel size plus a padding with the smallest possible value that leads to a multiple of 4 bytes (32 bits), which is the minimum padding accepted by [EDepthMap8::SetBufferPtr](#).

EDepthMap8.SetPixel

Sets the value of a pixel.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetPixel(
    Euresys.Open_eVision_2_10.EDepth8 value,
    int x,
    int y
)
```

Parameters

value

Value of the pixel.

x

Column of the pixel.

y

Row of the pixel.

EDepthMap8.SetSize

Sets the width and height of a DepthMap.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetSize(
    int width,
    int height
)
void SetSize(
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap other
)
```

Parameters

width

The new requested DepthMap width.

height

The new requested DepthMap height.

other

The other DepthMap whose dimensions have to be used for the current object.

Remarks

Open eVision will allocate a new buffer (deallocate the old buffer) if the supplied width and height are different from the existing ones.

If an external buffer has been specified by means of **SetImagePtr**, it will be kept only if the size does not change.

Creating a new buffer and setting its size creates a 4-byte aligned buffer, by default.

EDepthMap8.Type

Returns the image type.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override Euresys.Open_eVision_2_10.EImageType Type
```

```
{ get; }
```

EDepthMap8.UndefinedValue

Returns the Undefined value. That value is used to mark pixels with no valid depth value.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.EDepth8 UndefinedValue  
  
{ get; }
```

EDepthMap8.Width

Access depth map Width.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
override int Width  
  
{ get; set; }
```

EDepthMap8.ZResolution

Access the Z Resolution (depth units per grey scale value).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
override float ZResolution  
  
    { get; set; }
```

4.59. EDepthMapToMeshConverter Class

Performs the conversion from a [EDepthMap](#) to a [EMesh](#), using the given calibration model. A Depth Map is a grayscale image acquired by a laser triangulation system. The calibration model defines how to transform a pixel from the Depth Map to a world space position. The resulting 3D representation contains a [EMesh](#) representing the surface in the world space.

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

[CalibrationModel](#)

M Access the [ECalibrationModel](#) used for conversion.

e

Methods

[Convert](#)

Using the [ECalibrationModel](#), the method 'Convert' performs the conversion from a [EDepthMap](#) to a [EMesh](#).

[EDepthMapToMeshConverter](#)

Creates an [EDepthMapToMeshConverter](#) object.

[Load](#)

Loads the converter configuration. The given [ESerializer](#) must have been created for reading.

operator=

Assignment operator.

Save

Saves the converter configuration. The given [ESerializer](#) must have been created for writing.

D

epthMapToMeshConverter.CalibrationModel

Access the [ECalibrationModel](#) used for conversion.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
Euresys.Open_eVision_2_10.Easy3D.ECalibrationModel CalibrationModel  
{ get; set; }
```

EDepthMapToMeshConverter.Convert

Using the [ECalibrationModel](#), the method 'Convert' performs the conversion from a [EDepthMap](#) to a [EMesh](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```

void Convert(
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 srcDepthMap,
    Euresys.Open_eVision_2_10.Easy3D.EMesh obj
)

void Convert(
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 srcDepthMap,
    Euresys.Open_eVision_2_10.Easy3D.EMesh obj
)

void Convert(
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 srcDepthMap,
    Euresys.Open_eVision_2_10.ERegion region,
    Euresys.Open_eVision_2_10.Easy3D.EMesh obj
)

void Convert(
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 srcDepthMap,
    Euresys.Open_eVision_2_10.ERegion region,
    Euresys.Open_eVision_2_10.Easy3D.EMesh obj
)

```

Parameters

srcDepthMap

The Depth Map to convert.

obj

The destination mesh.

region

The region of interest.

EDepthMapToMeshConverter.EDepthMapToMeshConverter

Creates an [EDepthMapToMeshConverter](#) object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

[C#]

```
void EDepthMapToMeshConverter (
)

void EDepthMapToMeshConverter (
    Euresys.Open_eVision_2_10.Easy3D.EDepthMapToMeshConverter other
)
```

Parameters

other

Another [EDepthMapToMeshConverter](#).

EDepthMapToMeshConverter.Load

Loads the converter configuration. The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]

void Load (
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The serializer.

EDepthMapToMeshConverter.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EDepthMapToMeshConverter operator=(
    Euresys.Open_eVision_2_10.Easy3D.EDepthMapToMeshConverter other
)
```

Parameters

other

-

EDepthMapToMeshConverter.Save

Saves the converter configuration. The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The serializer.

4.60. EDepthMapToPointCloudConverter Class

Performs the conversion from a [EDepthMap](#) to a [EPointCloud](#), using the given calibration model. A Depth Map is a grayscale image acquired by a laser triangulation system. The calibration model defines how to transform a pixel from the Depth Map to a world space position. The resulting [EPointCloud](#) contains a point per defined pixel of the Depth Map. Undefined pixels are discarded.

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

CalibrationModel

M Access the [ECalibrationModel](#) used for conversion.

e

Methods

Convert

Applies the [ECalibrationModel](#) to the Depth Map pixels and fill the [EPointCloud](#) with world positions.

EDepthMapToPointCloudConverter

Create a [EDepthMapToPointCloudConverter](#).

Load

Loads the converter configuration. The given [ESerializer](#) must have been created for reading.

operator=

Assignment operator.

Save

Saves the converter configuration. The given [ESerializer](#) must have been created for writing.

epthMapToPointCloudConverter.CalibrationModel

Access the [ECalibrationModel](#) used for conversion.

Namespace: Euresys.Open_eVision_2_10.Easy3D

[C#]

```
Euresys.Open_eVision_2_10.Easy3D.ECalibrationModel CalibrationModel  
{ get; set; }
```

EDepthMapToPointCloudConverter.Convert

Applies the [ECalibrationModel](#) to the Depth Map pixels and fill the [EPointCloud](#) with world positions.

Namespace: Euresys.Open_eVision_2_10.Easy3D

[C#]

```
void Convert(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 srcDepthMap,  
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud pc  
)
```

```

void Convert(
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 srcDepthMap,
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud pc
)

void Convert(
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap32f srcDepthMap,
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud pc
)

void Convert(
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 srcDepthMap,
    Euresys.Open_eVision_2_10.ERegion region,
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud pc
)

void Convert(
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 srcDepthMap,
    Euresys.Open_eVision_2_10.ERegion region,
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud pc
)

void Convert(
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap32f srcDepthMap,
    Euresys.Open_eVision_2_10.ERegion region,
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud pc
)

```

Parameters

srcDepthMap

The Depth Map to convert.

pc

The destination Point Cloud.

region

The region of interest, only pixels inside the given region are converted and added to the Point Cloud.

EDepthMapToPointCloudConverter.EDepthMapToPointCloudCo nverter

Create a [EDepthMapToPointCloudConverter](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void EDepthMapToPointCloudConverter (
)
void EDepthMapToPointCloudConverter (
    Euresys.Open_eVision_2_10.Easy3D.EDepthMapToPointCloudConverter other
)
```

Parameters

other

Another [EDepthMapToPointCloudConverter](#).

EDepthMapToPointCloudConverter.Load

Loads the converter configuration. The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Load (
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The serializer.

EDepthMapToPointCloudConverter.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EDepthMapToPointCloudConverter operator=(
    Euresys.Open_eVision_2_10.Easy3D.EDepthMapToPointCloudConverter other
)
```

Parameters

other

Another [EDepthMapToPointCloudConverter](#).

EDepthMapToPointCloudConverter.Save

Saves the converter configuration. The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The serializer.

4.61. EEllipseRegion Class

Manages a complete context for an [ERegion](#) shaped like an ellipse.

Base Class: [ERegion](#)

Namespace: Euresys.Open_eVision_2_10

Properties

[Angle](#)

Angle of the region

[Center](#)

Center of the region

[HorizontalRadius](#)

Horizontal radius of the region

[VerticalRadius](#)

M Vertical radius of the region

e

Methods

[Drag](#)

Moves the specified handle to a new position and updates all placement parameters of the region.

[EEllipseRegion](#)

Constructs an [EEllipseRegion](#) context.

[HitTest](#)

Detects if the cursor is placed over one of the dragging handles.

[Load](#)

Loads the [EEllipseRegion](#). The given [ESerializer](#) must have been created for reading.

[operator!=](#)

Checks if this [EEllipseRegion](#) instance is not strictly equal to another

operator=

Assignment operator.

operator==

Checks if this [EEllipseRegion](#) instance is strictly equal to another

Rotate

Creates a new [EEllipseRegion](#) by rotating the [EEllipseRegion](#).

Save

Saves the [EEllipseRegion](#). The given [ESerializer](#) must have been created for writing.

Scale

Creates a new [EEllipseRegion](#) by scaling the [EEllipseRegion](#).

Translate

E Creates a new [EEllipseRegion](#) by translating the [EEllipseRegion](#).

E

EllipseRegion.Angle

Angle of the region

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float Angle
```

```
{ get; set; }
```

EEllipseRegion.Center

Center of the region

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint Center
{ get; set; }
```

EEllipseRegion.Drag

Moves the specified handle to a new position and updates all placement parameters of the region.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Drag(
    int x,
    int y,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)
```

Parameters

x
x-coordinate of the mouse cursor.

y
y-coordinate of the mouse cursor.

zoomX

Horizontal zoom factor. By default, true scale is used.

zoomY

Vertical zoom factor. By default, true scale is used.

panX

Horizontal pan offset. By default, no pan is added.

panY

Vertical pan offset. By default, no pan is added.

Remarks

If zooming and/or panning were used when drawing the region, the same values must be used with [EEllipseRegion::HitTest](#) and [EEllipseRegion::Drag](#).

EEllipseRegion.EEllipseRegion

Constructs an [EEllipseRegion](#) context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EEllipseRegion(
)
void EEllipseRegion(
    float centerX,
    float centerY,
    float radius1,
    float radius2,
    float angle
)
void EEllipseRegion(
    Euresys.Open_eVision_2_10.EPoint center,
    float radius1,
    float radius2,
    float angle
)
```



```

void EEllipseRegion(
    Euresys.Open_eVision_2_10.EPoint center,
    Euresys.Open_eVision_2_10.EPoint axisEnd1,
    Euresys.Open_eVision_2_10.EPoint axisEnd2
)

void EEllipseRegion(
    Euresys.Open_eVision_2_10.EPoint pt1,
    Euresys.Open_eVision_2_10.EPoint pt2,
    Euresys.Open_eVision_2_10.EPoint pt3,
    Euresys.Open_eVision_2_10.EPoint pt4,
    Euresys.Open_eVision_2_10.EPoint pt5
)

void EEllipseRegion(
    Euresys.Open_eVision_2_10.EEllipseRegion other
)

```

Parameters

centerX

The abscissa of the center of the [EEllipseRegion](#).

centerY

The ordinate of the center of the [EEllipseRegion](#).

radius1

The abscissa radius of the non rotated [EEllipseRegion](#).

radius2

The ordinate radius of the non rotated [EEllipseRegion](#).

angle

The angle of the rotated [EEllipseRegion](#).

center

The center of the [EEllipseRegion](#).

axisEnd1

The point corresponding to the end of the abscissa axis of the non rotated [EEllipseRegion](#).

axisEnd2

The point corresponding to the end of the ordinate axis of the non rotated [EEllipseRegion](#).

pt1

One of the five points defining the [EEllipseRegion](#).

pt2

One of the five points defining the [EEllipseRegion](#).

pt3

One of the five points defining the [EEllipseRegion](#).

pt4

One of the five points defining the [EEllipseRegion](#).

pt5

One of the five points defining the [EEllipseRegion](#).

other

[EEllipseRegion](#) context to copy.

Remarks

When defining an [EEllipseRegion](#), the two resulting radius values must not be **0** else an [Parameter3OutOfRange](#) or [EError](#) is thrown.

When defining an [EEllipseRegion](#), the two resulting radius valued must be small enough so that the region fit in memory.

When defining an [EEllipseRegion](#) with two axis ends, the two axis ends and the center must not all lie on the same straight line else an [EError](#) is thrown.

When defining an [EEllipseRegion](#) with five points, no more than two of the points should lie on the same straight line else an [EError](#) is thrown.

EEllipseRegion.HitTest

Detects if the cursor is placed over one of the dragging handles.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EEditionMode HitTest(
    int x,
    int y,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)
```

Parameters

x

x-coordinate of the mouse cursor.

y

y-coordinate of the mouse cursor.

zoomX

Horizontal zoom factor. By default, true scale is used.

zoomY

Vertical zoom factor. By default, true scale is used.

panX

Horizontal pan offset. By default, no pan is added.

panY

Vertical pan offset. By default, no pan is added.

Remarks

Returns a handle identifier, as defined by [EEditionMode](#).

If zooming and/or panning were used when drawing the region, the same values must be used with [EEllipseRegion::HitTest](#) and [EEllipseRegion::Drag](#).

EEllipseRegion.HorizontalRadius

Horizontal radius of the region

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float HorizontalRadius  
  
{ get; set; }
```

EEllipseRegion.Load

Loads the [EEllipseRegion](#). The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Load(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The serializer.

EEllipseRegion.operator!=

Checks if this [EEllipseRegion](#) instance is not strictly equal to another

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool operator!=(
    Euresys.Open_eVision_2_10.EEllipseRegion other
)
```

Parameters

other

Reference to the other [EEllipseRegion](#) instance

EEllipseRegion.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EEllipseRegion operator=(
    Euresys.Open_eVision_2_10.EEllipseRegion other
)
```

Parameters

other

Reference to the [EEllipseRegion](#) used for the assignment

EEllipseRegion.operator==

Checks if this [EEllipseRegion](#) instance is strictly equal to another

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool operator==(
    Euresys.Open_eVision_2_10.EEllipseRegion other
)
```

Parameters

other

Reference to the other [EEllipseRegion](#) instance

EEllipseRegion.Rotate

Creates a new [EEllipseRegion](#) by rotating the [EEllipseRegion](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EEllipseRegion Rotate(
    float angle
)
```

Parameters

angle
Rotation angle

EEllipseRegion.Save

Saves the [EEllipseRegion](#). The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer
The serializer.

EEllipseRegion.Scale

Creates a new [EEllipseRegion](#) by scaling the [EEllipseRegion](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EEllipseRegion Scale(
    float scale
)

Euresys.Open_eVision_2_10.EEllipseRegion Scale(
    float scaleX,
    float scaleY
)
```

Parameters

scale

Isotropic scale

scaleX

Horizontal scale

scaleY

Vertical scale

EEllipseRegion.Translate

Creates a new [EEllipseRegion](#) by translating the [EEllipseRegion](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EEllipseRegion Translate(
    float dx,
    float dy
)
```

Parameters

dx

Horizontal translation in pixel value

dy

Vertical translation in pixel value

EEllipseRegion.VerticalRadius

Vertical radius of the region

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float VerticalRadius  
  
{ get; set; }
```

4.62. EErrorStatistics Class

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

Max	Gets/Sets the maximum error (calculated on the valid samples).
Mean	Gets/Sets the mean error (calculated on the valid samples).
Min	Gets/Sets the minimum error (calculated on the valid samples).
NumOfErrors	Gets/Sets the number of errors (samples not found) which is the number of samples on which no error could be calculated.

NumOfValidSamples

Gets/Sets the number of valid samples which is the number of samples on which the error is calculated.

StdDev

M Gets/Sets the standard deviation error (calculated on the valid samples).

e

Methods

CopyTo

-

EErrorStatistics

Creates a [EErrorStatistics](#) object.

Load

Loads a [EErrorStatistics](#). The given [ESerializer](#) must have been created for reading.

operator=

Assignment operator.

Save

E Saves a [EErrorStatistics](#). The given [ESerializer](#) must have been created for writing.

E

EErrorStatistics.CopyTo

-

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void CopyTo(
    Euresys.Open_eVision_2_10.Easy3D.EErrorStatistics other
)
```

Parameters

other

-

EErrorStatistics.EErrorStatistics

Creates a [EErrorStatistics](#) object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void EErrorStatistics(
)
void EErrorStatistics(
    Euresys.Open_eVision_2_10.Easy3D.EErrorStatistics other
)
```

Parameters

other

Reference used for the initialization.

EErrorStatistics.Load

Loads a [EErrorStatistics](#). The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Load(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

-

EErrorStatistics.Max

Gets/Sets the maximum error (calculated on the valid samples).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
float Max
{ get; set; }
```

EErrorStatistics.Mean

Gets/Sets the mean error (calculated on the valid samples).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
float Mean
{ get; set; }
```

EErrorStatistics.Min

Gets/Sets the minimum error (calculated on the valid samples).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float Min  
{ get; set; }
```

EErrorStatistics.NumOfErrors

Gets/Sets the number of errors (samples not found) which is the number of samples on which no error could be calculated.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
int NumOfErrors  
{ get; set; }
```

EErrorStatistics.NumOfValidSamples

Gets/Sets the number of valid samples which is the number of samples on which the error is calculated.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
int NumOfValidSamples
{ get; set; }
```

EErrorStatistics.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EErrorStatistics operator=(
    Euresys.Open_eVision_2_10.Easy3D.EErrorStatistics other
)
```

Parameters

other

-

EErrorStatistics.Save

Saves a [EErrorStatistics](#). The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
void Save(  
    Euresys.Open_eVision_2_10.ESerializer serializer  
)
```

Parameters

serializer

-

EErrorStatistics.StdDev

Gets/Sets the standard deviation error (calculated on the valid samples).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float StdDev  
    { get; set; }
```

4.63. EException Class

Holds the exception information, that is the code and the description of the error that has thrown the exception.

Remarks

Each time an Open eVision error occurs, an exception is thrown. Exceptions feature an error code and a description. To catch a potentially arising exception, the function call is included in a try-catch block.

Namespace: Euresys.Open_eVision_2_10

Properties

Error

M

e

Methods

EException

-

operator=

-

What

E Returns the description of the error that has thrown the exception.
E

Exception.EException

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EException(
)
void EException(
    Euresys.Open_eVision_2_10.EError error
)
void EException(
    Euresys.Open_eVision_2_10.EException other
)
```

```
void EException(  
    string message  
)
```

Parameters

error

-

other

-

message

-

EException.Error

-

Namespace: Euresys.Open_eVision_2_10

[C#]

```
Euresys.Open_eVision_2_10.EError Error
```

```
{ get; set; }
```

EException.operator=

-

Namespace: Euresys.Open_eVision_2_10


```
[C#]
Euresys.Open_eVision_2_10.EException operator=(
    Euresys.Open_eVision_2_10.EException other
)
```

Parameters

other

-

EException.What

Returns the description of the error that has thrown the exception.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
string What(
)
```

4.64. EExplicitGeometricCalibrationModel Class

[EExplicitGeometricCalibrationModel](#) is used to calibrate a depth map from a minimal set of explicit geometric values. All model parameters are given to the [EExplicitGeometricCalibrationModel](#) constructor.

Base Class: [ECalibrationModel](#)

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

CameraAngle

Camera view angle.

CameraHeight

The height from the camera optical center to the object reference plane (assuming the reference plane is projected in the center line of the image), in mm.

FocalLength

Camera focal length.

LaserPlaneAngle

Laser plane angle.

MotionIncrement

Motion increment (Distance between each line of the depth map).

RoiBottomLine

The ROI bottom line used in laser line extraction.

RoiLeftColumn

The ROI left column used in laser line extraction.

SensorHeight

Camera sensor height.

SensorWidth

Camera sensor width.

SensorXResolution

Camera X resolution.

SensorYResolution

Camera Y resolution.

Type

M Returns the type of calibration model, see [ECalibrationType](#).

e

Methods

EExplicitGeometricCalibrationModel

Constructs a [EExplicitGeometricCalibrationModel](#).
[EExplicitGeometricCalibrationModel](#) is used to calibrate a depth map from a minimal set of explicit geometric values.

IsInitialized

Returns true if the model has been correctly initialized.

Load

Loads the [EExplicitGeometricCalibrationModel](#). The given [ESerializer](#) must have been created for reading.

operator=

Assignment operator.

operator==

Comparison operator.

Save

E Saves the [EExplicitGeometricCalibrationModel](#). The given [ESerializer](#) must have been created for writing.

E

ExplicitGeometricCalibrationModel.CameraAngle

Camera view angle.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float CameraAngle  
    { get; }
```

EExplicitGeometricCalibrationModel.CameraHeight

The height from the camera optical center to the object reference plane (assuming the reference plane is projected in the center line of the image), in mm.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float CameraHeight  
    { get; }
```

EExplicitGeometricCalibrationModel.EExplicitGeometricCalibrationModel

Constructs a [EExplicitGeometricCalibrationModel](#).
[EExplicitGeometricCalibrationModel](#) is used to calibrate a depth map from a minimal set of explicit geometric values.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```

void EExplicitGeometricCalibrationModel (
)

void EExplicitGeometricCalibrationModel (
    float sensorWidth,
    float sensorHeight,
    int sensorXResolution,
    int sensorYResolution,
    int roiLeftColumn,
    int roiBottomLine,
    float focalLength,
    float cameraAngle,
    float cameraHeight,
    float laserPlaneAngle,
    float motionIncrement
)

void EExplicitGeometricCalibrationModel (
    Euresys.Open_eVision_2_10.Easy3D.EExplicitGeometricCalibrationModel other
)

```

Parameters

sensorWidth

The camera sensor width, in mm.

sensorHeight

The camera sensor height, in mm.

sensorXResolution

The camera X resolution (pixel count in width).

sensorYResolution

The camera Y resolution (pixel count in height).

roiLeftColumn

The ROI left column used in laser line extraction.
Between the left (0) and the right (width) of the image.

roiBottomLine

The ROI bottom line used in laser line extraction.
Between the top (0) and the bottom (height) of the image. That's the depth map values origin.

focalLength

The camera optics focal length, in mm.

cameraAngle

The camera angle from the vertical axis.

Looking down camera is angle 0 and positive in counter clockwise direction. Valid values are between 0 (vertical orientation) and lower than 90° (horizontal orientation).

cameraHeight

The height from the camera optical center to the object reference plane (assuming the reference plane is projected in the center line of the image), in mm.

laserPlaneAngle

The laser plane angle from the vertical axis.

A perfect vertical laser orientation is angle 0 and negative in clockwise direction. Valid values for laser angle are between -90° (excluded) and +90° (excluded).

motionIncrement

The distance in mm between each line of the depth map.

That's the relative motion of the camera/laser setup to the object position.

other

Another [EExplicitGeometricCalibrationModel](#).

EExplicitGeometricCalibrationModel.FocalLength

Camera focal length.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
float FocalLength  
{ get; }
```

EExplicitGeometricCalibrationModel.IsInitialized

Returns true if the model has been correctly initialized.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool IsInitialized(
)
```

EExplicitGeometricCalibrationModel.LaserPlaneAngle

Laser plane angle.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
float LaserPlaneAngle
{ get; }
```

EExplicitGeometricCalibrationModel.Load

Loads the [EExplicitGeometricCalibrationModel](#). The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Load(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The serializer.

EExplicitGeometricCalibrationModel.MotionIncrement

Motion increment (Distance between each line of the depth map).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float MotionIncrement  
  
    { get; }
```

EExplicitGeometricCalibrationModel.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.Easy3D.EExplicitGeometricCalibrationModel operator=(  
    Euresys.Open_eVision_2_10.Easy3D.EExplicitGeometricCalibrationModel other  
)
```

Parameters

other

Another [EExplicitGeometricCalibrationModel](#).

EExplicitGeometricCalibrationModel.operator==

Comparison operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool operator==(
    Euresys.Open_eVision_2_10.Easy3D.EExplicitGeometricCalibrationModel other
)
```

Parameters

other

The other [EExplicitGeometricCalibrationModel](#).

EExplicitGeometricCalibrationModel.RoiBottomLine

The ROI bottom line used in laser line extraction.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
int RoiBottomLine
{ get; }
```

EExplicitGeometricCalibrationModel.RoiLeftColumn

The ROI left column used in laser line extraction.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
int RoiLeftColumn
{ get; }
```

EExplicitGeometricCalibrationModel.Save

Saves the [EExplicitGeometricCalibrationModel](#). The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The serializer.

EExplicitGeometricCalibrationModel.SensorHeight

Camera sensor height.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
float SensorHeight
```

```
{ get; }
```

EExplicitGeometricCalibrationModel.SensorWidth

Camera sensor width.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float SensorWidth  
    { get; }
```

EExplicitGeometricCalibrationModel.SensorXResolution

Camera X resolution.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
int SensorXResolution  
    { get; }
```

EExplicitGeometricCalibrationModel.SensorYResolution

Camera Y resolution.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
int SensorYResolution
{ get; }
```

EExplicitGeometricCalibrationModel.Type

Returns the type of calibration model, see [ECalibrationType](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override Euresys.Open_eVision_2_10.Easy3D.ECalibrationType Type
{ get; }
```

4.65. EFeaturesAligner Class

Alignment class, used to calculate the best transformation between matching pairs of points.

The object [EFeaturesAligner](#) contains a list of points called the 'Model points list'.

The method [EFeaturesAligner::Compute](#) takes a second list of points as argument which is called the 'Measured points list' and produces a [E3DTransformMatrix](#) as result.

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

ModelPoints

Sets/Gets the model points list.

The model point list is a list of points that is used either as source or as destination of the transformation to calculate.

The model points list should at least contain 3 unaligned points.

PolarityTransform

M Sets/Gets the polarity of the transformation from [EAlignmentPolarity](#).

e

Methods

Compute

Computes the best transformation matrix for a given Measured points list.
This will compute the best transformation for the alignment between the Measured points and the Model points.

EFeaturesAligner

Creates a [EFeaturesAligner](#) object.

Load

Loads the [EFeaturesAligner](#) object configuration. The given [ESerializer](#) must have been created for reading.

operator=

Assignment operator.

Save

E Saves the [EFeaturesAligner](#) object configuration. The given [ESerializer](#) must have been created for writing.

F

FeaturesAligner.Compute

Computes the best transformation matrix for a given Measured points list.
This will compute the best transformation for the alignment between the Measured points and the Model points.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix Compute(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint[] measuredPoints
)

Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix Compute(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint[] measuredPoints,
    Euresys.Open_eVision_2_10.Easy3D.EErrorStatistics errorStatistics
)
```

Parameters

measuredPoints

The measured points list; the measured points list should at least contain 3 unaligned points.

errorStatistics

Optional parameter passed by reference. This object will contains the error statistics (deviation between model and aligned points).

EFeaturesAligner.EFeaturesAligner

Creates a [EFeaturesAligner](#) object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void EFeaturesAligner(
)

void EFeaturesAligner(
    Euresys.Open_eVision_2_10.Easy3D.EFeaturesAligner other
)
```

Parameters

other

Reference to the object used for the initialization.

EFeaturesAligner.Load

Loads the [EFeaturesAligner](#) object configuration. The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Load(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The serializer.

EFeaturesAligner.ModelPoints

Sets/Gets the model points list.

The model point list is a list of points that is used either as source or as destination of the transformation to calculate.

The model points list should at least contain 3 unaligned points.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DPoint[] ModelPoints
{ get; set; }
```

EFeaturesAligner.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EFeaturesAligner operator=(
    Euresys.Open_eVision_2_10.Easy3D.EFeaturesAligner other
)
```

Parameters

other

-

EFeaturesAligner.PolarityTransform

Sets/Gets the polarity of the transformation from [EAlignmentPolarity](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EAlignmentPolarity PolarityTransform
    { get; set; }
```

Remarks

If polarity is [ModelToMeasured](#), calculates the best transformation from the Model points list to the Measured points list (default).

If the polarity is [MeasuredToModel](#), calculates the best transformation from the Measured points list to the Model points list.

EFeaturesAligner.Save

Saves the [EFeaturesAligner](#) object configuration. The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The serializer.

4.66. EFilePointerSerializer Class

Base Class: [ESerializer](#)

Namespace: Euresys.Open_eVision_2_10

Properties

Writing

M Returns **TRUE** if the [ESerializer](#) object has been created for writing and **FALSE** otherwise.

Methods

Close

EFilePointerSerializer.Close

Closes the file associated with the [ESerializer](#) object.

e

Closes the file associated with the [ESerializer](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Close(  
)
```

EFilePointerSerializer.Writing

Returns **TRUE** if the [ESerializer](#) object has been created for writing and **FALSE** otherwise.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
override bool Writing  
{ get; }
```

4.67. EFileSerializer Class

-

Base Class: [ESerializer](#)

Namespace: Euresys.Open_eVision_2_10

Properties

Writing

M Returns **TRUE** if the **ESerializer** object has been created for writing and **FALSE** otherwise.

e

Methods

Close

E Closes the file associated with the **ESerializer** object.

F

FileSerializer.Close

Closes the file associated with the **ESerializer** object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Close(  
)
```

EFileSerializer.Writing

Returns **TRUE** if the **ESerializer** object has been created for writing and **FALSE** otherwise.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
override bool Writing
```

```
{ get; }
```

4.68. EFilters Class

Filtering functions used to remove noise on 3D containers.

Namespace: Euresys.Open_eVision_2_10.Easy3D

Methods

RemoveNoise

ilters.RemoveNoise

In a [EDepthMap](#) or [EZMap](#), removes noisy pixels. A square filter window is moved over the depthmap. Within this filter window, the average and/or the standard deviation is/are calculated and a rejection criterium defined by the parameter 'method' determines which pixels will be removed. If the rejection criterium determines that a pixel has to be removed or if there is not enough valid pixels in the window filter, as specified by the parameter 'minValidRatio' the pixel is either

In a [EDepthMap](#) or [EZMap](#), removes noisy pixels. A square filter window is moved over the depthmap. Within this filter window, the average and/or the standard deviation is/are calculated and a rejection criterium defined by the parameter 'method' determines which pixels will be removed.

If the rejection criterium determines that a pixel has to be removed or if there is not enough valid pixels in the window filter, as specified by the parameter 'minValidRatio', the pixel is either simply removed (marked 'invalid') or replaced by the average value (within the filter window), depending on the value of 'replaceByAvg'.

Namespace: Euresys.Open_eVision_2_10.Easy3D

[C#]

```
void RemoveNoise(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 sourceDepthMap,  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 destinationDepthMap,  
    Euresys.Open_eVision_2_10.Easy3D.ENoiseRemovalMethod method,  
    short halfKernelSize,  
    float threshold,  
    float minValidRatio,  
    bool replaceByAvg  
)
```

```
void RemoveNoise(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 sourceDepthMap,  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 destinationDepthMap,  
    Euresys.Open_eVision_2_10.Easy3D.ENoiseRemovalMethod method,  
    short halfKernelSize,  
    float threshold,  
    float minValidRatio,  
    bool replaceByAvg  
)
```

```
void RemoveNoise(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 sourceZMap,  
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 destinationZMap,  
    Euresys.Open_eVision_2_10.Easy3D.ENoiseRemovalMethod method,  
    short halfKernelSize,  
    float threshold,  
    float minValidRatio,  
    bool replaceByAvg  
)
```

```
void RemoveNoise(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 sourceZMap,  
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 destinationZMap,  
    Euresys.Open_eVision_2_10.Easy3D.ENoiseRemovalMethod method,  
    short halfKernelSize,  
    float threshold,  
    float minValidRatio,  
    bool replaceByAvg  
)
```

Parameters

sourceDepthMap

The source depthmap.

destinationDepthMap

The destination depthmap. It should have the same dimensions as the source depthmap.

method

Noise removal method of type [ENoiseRemovalMethod](#).

halfKernelSize

The half-size of the window filter. The filter window size (= kernel size) is $\text{halfKernelSize} * 2 + 1$, should be positive, smaller than (or equal to) the image size, and may not exceed 256.

threshold

The threshold used by the rejection criterium.

minValidRatio

Required ratio of valid pixels in the filter window to process the calculation. If not enough, marks the pixel for replacement. Setting this ratio to 0.0 means that only one pixel has to be valid. The default value is 0.25 .

replaceByAvg

The marked pixels are removed by default; if this parameter is set to true, replaces the marked pixels by the average value, calculated within the filter window.

sourceZMap

The source ZMap.

destinationZMap

The destination ZMap. It should have the same dimensions as the source ZMap.

4.69. EFloatRange Class

Represents a range of floating point values.

Namespace: Euresys.Open_eVision_2_10

Properties

Center

Center of the range.

LowerBound

Lower bound of the range.

Size

Size of the range.

UpperBound

M Upper bound of the range.

e

thods

EFloatRange

Creates an [EFloatRange](#) object.

IsInRange

Checks if a value is inside the range.

operator=

Assignment operator

operator==

Comparison operator

SetBounds

Sets the bounds of the range.

SetFromBaseAndAbsoluteTolerance

Sets the bounds of the range from a base value and an absolute tolerance from this base value.

SetFromBaseAndRelativeTolerance

Sets the bounds of the range from a base value and a relative tolerance from this base value.

Update

Updates the range. If the value passed is outside of the range bounds, they are modified so they include the value.

EFloatRange.Center

Center of the range.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float Center
{ get; }
```

EFloatRange.EFloatRange

Creates an [EFloatRange](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EFloatRange (
)
void EFloatRange (
    float min,
    float max
)
void EFloatRange (
    Euresys.Open_eVision_2_10.EFloatRange range
)
```

Parameters

min

Lower bound of the range.

max

Upper bound of the range.

range

Range to copy.

EFloatRange.IsInRange

Checks if a value is inside the range.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool IsInRange (
    float value,
    bool lowerBoundInclusive,
    bool upperBoundInclusive
)
```

Parameters

value

Value to test.

lowerBoundInclusive

Indicates if the lower bound should be considered inside the range (true) or outside (false).

upperBoundInclusive

Indicates if the upper bound should be considered inside the range (true) or outside (false).

EFloatRange.LowerBound

Lower bound of the range.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float LowerBound  
{ get; }
```

EFloatRange.operator=

Assignment operator

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EFloatRange operator=(  
    Euresys.Open_eVision_2_10.EFloatRange other  
)
```

Parameters

other

-

EFloatRange.operator==

Comparison operator

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
bool operator==(
    Euresys.Open_eVision_2_10.EFloatRange other
)
```

Parameters

other

The other object.

EFloatRange.SetBounds

Sets the bounds of the range.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetBounds(
    float min,
    float max
)
```

Parameters

min

Lower bound of the range.

max

Upper bound of the range.

EFloatRange.SetFromBaseAndAbsoluteTolerance

Sets the bounds of the range from a base value and an absolute tolerance from this base value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetFromBaseAndAbsoluteTolerance (
    float baseValue,
    float tolerance
)
```

Parameters

baseValue

Base value.

tolerance

Absolute tolerance around the base value. Must be positive.

Remarks

The range will be set with a lower bound of (base - tolerance) and an upper bound of (base + tolerance).

EFloatRange.SetFromBaseAndRelativeTolerance

Sets the bounds of the range from a base value and a relative tolerance from this base value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetFromBaseAndRelativeTolerance (
    float baseValue,
    float tolerance
)
```

Parameters

baseValue

Base value.

tolerance

Relative tolerance around the base value. Must be positive.

Remarks

The range will be set with a lower bound of (base - (base * tolerance)) and an upper bound of (base + (base * tolerance)).

EFloatRange.Size

Size of the range.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Size  
    { get; }
```

EFloatRange.Update

Updates the range. If the value passed is outside of the range bounds, they are modified so they include the value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Update(  
    float value  
)
```

Parameters

value

Value to be included in the range.

EFloatRange.UpperBound

Upper bound of the range.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float UpperBound  
  
    { get; }
```

4.70. EFoundPattern Class

Represents a single instance of the pattern in the search field, as returned by the EasyFind finding process.

Remarks

[EPatternFinder::Find](#) returns a collection of instances of this class. An EFoundPattern object represents one found instance, with all the needed information about it.

Namespace: Euresys.Open_eVision_2_10

Properties

Angle	Angle of the found instance, in the current angle unit.
Center	Reference point of the instance.
DrawBoundingBox	Flag indicating if the EFoundPattern::Draw method must draw the bounding box of the FoundPattern .
DrawCenter	Flag indicating if the EFoundPattern::Draw method must draw the center of the FoundPattern .

DrawFeaturePoints

Flag indicating if the [EFoundPattern::Draw](#) method must draw the feature points of the [EFoundPattern](#) object.

Quadrangle

Returns the corners of the bounding box of the found pattern.

Scale

Scaling factor of the found pattern, in units (not percents).

Score

M Matching score of the found pattern, in units (not percents).

e

Methods

Draw

Draws the found pattern, in image coordinates.

DrawWithCurrentPen

Draws the found pattern, in image coordinates.

EFoundPattern

Constructs a [EFoundPattern](#) object.

operator!=

Inequality operator.

operator=

Copies all the data from another [EFoundPattern](#) object into the current [EFoundPattern](#) object

operator==

Equality operator.

EFoundPattern.Angle

Angle of the found instance, in the current angle unit.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Angle  
    { get; }
```

Remarks

Read-only. This returned value is always comprised in the range [- a half turn, + a half turn].

EFoundPattern.Center

Reference point of the instance.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint Center  
    { get; }
```

Remarks

By default, this is its center. If the property [EPatternFinder::Pivot](#) has been changed in the [EPatternFinder](#), the point returns the pivot in the instance.

EFoundPattern.Draw

Draws the found pattern, in image coordinates.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Draw(
    IntPtr graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

Parameters

graphicContext

Handle to the device context of the destination window.

zoomX

Horizontal zooming factor.

zoomY

Vertical zooming factor. If set to **0**, the horizontal zooming factor will be used for isotropic zooming.

panX

Horizontal panning factor.

panY

Vertical panning factor.

color

The color in which to draw the overlay.

Remarks

This method draws different features of the `EFoundPattern`, according to the value of properties `EFoundPattern::DrawFeaturePoints`, `EFoundPattern::DrawCenter`, `EFoundPattern::DrawBoundingBox`. The **zoomX**, **zoomY**, **panX** and **panY** parameters can be used to scale and/or translate the drawing operations.

EFoundPattern.DrawBoundingBox

Flag indicating if the `EFoundPattern::Draw` method must draw the bounding box of the **FoundPattern**.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
bool DrawBoundingBox  
  
{ get; set; }
```

Remarks

The default value is **TRUE**.

EFoundPattern.DrawCenter

Flag indicating if the `EFoundPattern::Draw` method must draw the center of the **FoundPattern**.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool DrawCenter
{ get; set; }
```

Remarks

The default value is **TRUE**.

EFoundPattern.DrawFeaturePoints

Flag indicating if the [EFoundPattern::Draw](#) method must draw the feature points of the [EFoundPattern](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool DrawFeaturePoints
{ get; set; }
```

Remarks

The default value is **FALSE**.

EFoundPattern.DrawWithCurrentPen

Draws the found pattern, in image coordinates.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void DrawWithCurrentPen(  
    IntPtr graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

Parameters

graphicContext

Handle to the device context of the destination window.

zoomX

Horizontal zooming factor.

zoomY

Vertical zooming factor. If set to **0**, the horizontal zooming factor will be used for isotropic zooming.

panX

Horizontal panning factor.

panY

Vertical panning factor.

Remarks

This method draws different features of the `EFoundPattern`, according to the value of properties [EFoundPattern::DrawFeaturePoints](#), [EFoundPattern::DrawCenter](#), [EFoundPattern::DrawBoundingBox](#). The **zoomX**, **zoomY**, **panX** and **panY** parameters can be used to scale and/or translate the drawing operations.

EFoundPattern.EFoundPattern

Constructs a `EFoundPattern` object.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void EFoundPattern(  
    )  
  
void EFoundPattern(  
    Euresys.Open_eVision_2_10.EFoundPattern other  
    )
```

Parameters

other

EFoundPattern object to be copied

EFoundPattern.operator!=

Inequality operator.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
bool operator!=(  
    Euresys.Open_eVision_2_10.EFoundPattern other  
    )
```

Parameters

other

Instance to compare to.

EFoundPattern.operator=

Copies all the data from another EFoundPattern object into the current EFoundPattern object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EFoundPattern operator=(
    Euresys.Open_eVision_2_10.EFoundPattern other
)
```

Parameters

other

EFoundPattern object to be copied

EFoundPattern.operator==

Equality operator.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool operator==(
    Euresys.Open_eVision_2_10.EFoundPattern other
)
```

Parameters

other

Instance to compare to.

EFoundPattern.Quadrangle

Returns the corners of the bounding box of the found pattern.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EQuadrangle Quadrangle  
  
{ get; }
```

EFoundPattern.Scale

Scaling factor of the found pattern, in units (not percents).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float Scale  
  
{ get; }
```

EFoundPattern.Score

Matching score of the found pattern, in units (not percents).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float Score  
  
{ get; }
```

Remarks

The matching score range is **[-1.0..1.0]**.

4.71. EFrame Class

Represents a geometrical frame of reference as well as the parameters needed to transform from/to local and global coordinates. It contains a point and an angle and serves as a base class for geometrical elements.

Base Class: [EPoint](#)

Derived Class(es): [ECircle](#) [ELine](#) [ERectangle](#) [EWedge](#)

Namespace: Euresys.Open_eVision_2_10

Properties

[Angle](#)

Orientation of the frame.

[CenterX](#)

Abscissa of the origin point of the frame.

[CenterY](#)

Ordinate of the origin point of the frame.

[Scale](#)

M Horizontal sensor resolution, in pixels per unit.

e

Methods

[CopyTo](#)

Copies all the data from the current EFrame object into another EFrame object, and returns it.

[EFrame](#)

Constructs a EFrame object.

GlobalToLocal

Transforms a geometrical element from global to local coordinates (world to local).

LocalToGlobal

Transforms a geometrical element from local to global coordinates (local to world).

operator=

Copies all the data from another EFrame object into the current EFrame object.

rame.Angle

Orientation of the frame.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float Angle
```

```
{ get; set; }
```

EFrame.CenterX

Abscissa of the origin point of the frame.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float CenterX
{ get; }
```

EFrame.CenterY

Ordinate of the origin point of the frame.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float CenterY
{ get; }
```

EFrame.CopyTo

Copies all the data from the current EFrame object into another EFrame object, and returns it.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EFrame CopyTo(
    Euresys.Open_eVision_2_10.EFrame other
)
```

Parameters

other

Pointer to the EFrame object in which the current EFrame object data have to be copied.

Remarks

In case of a **NULL** pointer, a new EFrame object will be created and returned.

EFrame.EFrame

Constructs a EFrame object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EFrame (
)

void EFrame (
    float centerX,
    float centerY,
    float angle,
    float scale
)

void EFrame (
    Euresys.Open_eVision_2_10.EPoint center,
    float angle,
    float scale
)

void EFrame (
    Euresys.Open_eVision_2_10.EFrame frame
)
```

Parameters

centerX

Abscissa of the origin point of the frame.

centerY

Ordinate of the origin point of the frame.

angle

Orientation of the frame.

scale

Horizontal sensor resolution.

center

Coordinates of the origin point of the frame.

frame

Pre-existing EFrame object used by the copy constructor.

EFrame.GlobalToLocal

Transforms a geometrical element from global to local coordinates (world to local).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint GlobalToLocal(
    Euresys.Open_eVision_2_10.EPoint global
)
Euresys.Open_eVision_2_10.EFrame GlobalToLocal(
    Euresys.Open_eVision_2_10.EFrame global
)
```

Parameters

global

The element, expressed in global coordinates.

EFrame.LocalToGlobal

Transforms a geometrical element from local to global coordinates (local to world).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EPoint LocalToGlobal(  
    Euresys.Open_eVision_2_10.EPoint local  
)  
  
Euresys.Open_eVision_2_10.EFrame LocalToGlobal(  
    Euresys.Open_eVision_2_10.EFrame local  
)  
  
Euresys.Open_eVision_2_10.ELine LocalToGlobal(  
    Euresys.Open_eVision_2_10.ELine local  
)  
  
Euresys.Open_eVision_2_10.ECircle LocalToGlobal(  
    Euresys.Open_eVision_2_10.ECircle local  
)
```

Parameters

local

The element, expressed in local coordinates.

EFrame.operator=

Copies all the data from another EFrame object into the current EFrame object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
Euresys.Open_eVision_2_10.EFrame operator=(  
    Euresys.Open_eVision_2_10.EFrame frame  
)
```

Parameters

frame

EFrame object to be copied.

EFrame.Scale

Horizontal sensor resolution, in pixels per unit.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Scale  
  
{ get; set; }
```

4.72. EFrameShape Class

Manages a complete context for measuring frame shapes.

Remarks

This class allows the grouping of several gauges or other frames.

Base Class: [EShape](#)

Namespace: Euresys.Open_eVision_2_10

Properties

[Angle](#)

The angle of the frame.

[Center](#)

Origin point coordinates of the frame.

[CenterX](#)

Gets the origin point abscissa of the frame.

CenterY

Gets the origin point ordinate of the frame.

Scale

The scale of the frame.

SizeX

Frame X-axis length.

SizeY

Frame Y-axis length.

Type

M Type of the frame.

e

Methods

Closest

Find the daughter shape that is the closest to this shape. To retrieve the closest shape, use [EShape::ClosestShape](#).

CopyTo

Copy operator.

Drag

Moves a handle to a new position and updates the position parameters of the shape.

Draw

Draws a graphical representation of a shape, as defined by [EDrawingMode](#).

DrawWithCurrentPen

Draws a graphical representation of a shape, as defined by [EDrawingMode](#).

EFrameShape

Construct a [EFrameShape](#) object.

HitTest

Checks whether the cursor is positioned over a handle (**TRUE**) or not (**FALSE**).

operator=

Assignment operator.

Set

Sets the coordinates of the origin point and the orientation of the frame.

SetCenterXY

Sets the origin point coordinates of the frame.

SetSize

☐ Sets the frame size.

F

FrameShape.Angle

The angle of the frame.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float Angle
```

```
{ get; set; }
```


EFrameShape.Center

Origin point coordinates of the frame.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint Center  
    { get; set; }
```

EFrameShape.CenterX

Gets the origin point abscissa of the frame.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float CenterX  
    { get; }
```

EFrameShape.CenterY

Gets the origin point ordinate of the frame.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float CenterY
{ get; }
```

EFrameShape.Closest

Find the daughter shape that is the closest to this shape. To retrieve the closest shape, use [EShape::ClosestShape](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Closest(
)
```

EFrameShape.CopyTo

Copy operator.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EFrameShape CopyTo(
    Euresys.Open_eVision_2_10.EFrameShape other,
    bool recursive
)
```

Parameters

other

[EFrameShape](#) object to copy to.

recursive

TRUE if the daughter shapes have to be copied as well, **FALSE** otherwise.

EFrameShape.Drag

Moves a handle to a new position and updates the position parameters of the shape.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Drag(
    int cursorX,
    int cursorY
)
```

Parameters

cursorX

Current cursor coordinates.

cursorY

Current cursor coordinates.

EFrameShape.Draw

Draws a graphical representation of a shape, as defined by [EDrawingMode](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)

void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingMode

Indicates how the shape must be displayed, as defined by [EDrawingMode](#).

daughters

TRUE if the daughter shapes are to be displayed also.

color

The color in which to draw the overlay.

EFrameShape.DrawWithCurrentPen

Draws a graphical representation of a shape, as defined by [EDrawingMode](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawWithCurrentPen(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingMode

Indicates how the point location or model fitting gauge must be displayed, as defined by [EDrawingMode](#).

daughters

TRUE if the daughters gauges are to be displayed also.

EFrameShape.EFrameShape

Construct a [EFrameShape](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EFrameShape(
)

void EFrameShape(
    Euresys.Open_eVision_2_10.EFrameShape frameShape
)
```

Parameters

frameShape

Pre-existing [EFrameShape](#) object used by the copy constructor.

Remarks

With the default constructor, all parameters are initialized to their respective default value. With the copy constructor, the constructed frame shape measurement context is based on a pre-existing [EFrameShape](#) object. By default, the daughter shapes are also copied. Use the [EFrameShape::CopyTo](#) method to disable explicitly the daughter shapes copy.

EFrameShape.HitTest

Checks whether the cursor is positioned over a handle (**TRUE**) or not (**FALSE**).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool HitTest(
    bool daughter
)
```

Parameters

daughter

TRUE if the daughters shapes handles have to be considered as well.

EFrameShape.operator=

Assignement operator.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EFrameShape operator=(
    Euresys.Open_eVision_2_10.EFrameShape other
)
```

Parameters

other

[EFrameShape](#) object to copy.

Remarks

By default, the daughter shapes are also copied. Use the [EFrameShape::CopyTo](#) method to disable explicitly the daughter shapes copy.

EFrameShape.Scale

The scale of the frame.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float Scale  
{ get; set; }
```

EFrameShape.Set

Sets the coordinates of the origin point and the orientation of the frame.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void Set(  
    Euresys.Open_eVision_2_10.EPoint center,  
    float angle,  
    float scale  
)
```

Parameters

center

Coordinates of the origin point of the frame. The default value is **(0,0)**.

angle

Rotation angle of the frame. The default value is **0**.

scale

Horizontal sensor resolution, in pixels per unit

EFrameShape.SetCenterXY

Sets the origin point coordinates of the frame.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetCenterXY(
    float centerX,
    float centerY
)
```

Parameters

centerX

Abscissa of the origin point of the frame. Default value is **0**.

centerY

Ordinate of the origin point of the frame. Default value is **0**.

EFrameShape.SetSize

Sets the frame size.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
void SetSize(
    float sizeX,
    float sizeY
)
```

Parameters

sizeX

Frame X-axis length. The default value is **100**.

sizeY

Frame Y-axis length. By default, both axes have the same length.

Remarks

By default, both frame axis value are set to **100**, which means 100 pixels when the field of view is not calibrated and 100 "units" in case of a calibrated field of view.

EFrameShape.SizeX

Frame X-axis length.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float SizeX
{ get; }
```

Remarks

By default, both frame axis values are set to **100**, which means 100 pixels when the field of view is not calibrated, and 100 physical units in case of a calibrated field of view.

EFrameShape.SizeY

Frame Y-axis length.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float SizeY  
  
    { get; }
```

Remarks

By default, both frame axis values are set to **100**, which means 100 pixels when the field of view is not calibrated, and 100 physical units in case of a calibrated field of view.

EFrameShape.Type

Type of the frame.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
override Euresys.Open_eVision_2_10.EShapeType Type  
  
    { get; }
```

4.73. EGrabberDepthMap16 Class

The EGrabberDepthMap16 class is used to wrap an EGrabber buffer whose pixel type is "Coord3D_C16". It can be used in Open eVision processing as an EDepthMap16 object.

Base Class: [EDepthMap16](#)

Namespace: Euresys.Open_eVision_2_10.EGrabberBridge

Methods

[EGrabberDepthMap16](#)

Constructs an EGrabberDepthMap16.

GrabberDepthMap16.EGrabberDepthMap16

Constructs an EGrabberDepthMap16.

Namespace: Euresys.Open_eVision_2_10.EGrabberBridge

```
[C#]
void EGrabberDepthMap16 (
    BufferInfo infos
)
void EGrabberDepthMap16 (
    ref FormatConvert converter,
    BufferInfo infos
)
void EGrabberDepthMap16 (
    Euresys.Open_eVision_2_10.EGrabberBridge.EGrabberDepthMap16
)
```

Parameters

infos

Information pertaining to an EGrabber buffer

converter

EGrabber format converter

-

Remarks

The FormatConverter class allows automatic pixel format conversions. If no format converter is provided, the pixel format of the buffer must be "Coord3D_C16"

4.74. EGrabberDepthMap8 Class

The EGrabberDepthMap8 class is used to wrap an EGrabber buffer whose pixel type is "Coord3D_C8". It can be used in Open eVision processing as an EDepthMap8 object.

Base Class: [EDepthMap8](#)

Namespace: Euresys.Open_eVision_2_10.EGrabberBridge

Methods

[EGrabberDepthMap8](#)

E Constructs an EGrabberDepthMap8.

G

GrabberDepthMap8.EGrabberDepthMap8

Constructs an EGrabberDepthMap8.

Namespace: Euresys.Open_eVision_2_10.EGrabberBridge

```
[C#]
void EGrabberDepthMap8 (
    BufferInfo infos
)

void EGrabberDepthMap8 (
    ref FormatConvert converter,
    BufferInfo infos
)

void EGrabberDepthMap8 (
    Euresys.Open_eVision_2_10.EGrabberBridge.EGrabberDepthMap8
)
```

Parameters

infos

Information pertaining to an EGrabber buffer

converter

EGrabber format converter

-

Remarks

The FormatConverter class allows automatic pixel format conversions. If no format converter is provided, the pixel format of the buffer must be "Coord3D_C8"

4.75. EGrabberImageBW16 Class

The EGrabberImageBW16 class is used to wrap an EGrabber buffer whose pixel type is "Mono16". It can be used in Open eVision processing as an EImageBW16 object.

Base Class: [EImageBW16](#)

Namespace: Euresys.Open_eVision_2_10.EGrabberBridge

Methods

[EGrabberImageBW16](#)

Constructs an EGrabberImageBW16.

EGrabberImageBW16.EGrabberImageBW16

Constructs an EGrabberImageBW16.

Namespace: Euresys.Open_eVision_2_10.EGrabberBridge

```
[C#]
void EGrabberImageBW16 (
    BufferInfo infos
)
void EGrabberImageBW16 (
    ref FormatConvert converter,
    BufferInfo infos
)
void EGrabberImageBW16 (
    Euresys.Open_eVision_2_10.EGrabberBridge.EGrabberImageBW16
)
```

Parameters

infos

Information pertaining to an EGrabber buffer

converter

EGrabber format converter

-

Remarks

The FormatConverter class allows automatic pixel format conversions. If no format converter is provided, the pixel format of the buffer must be "Mono16"

4.76. EGrabberImageBW8 Class

The EGrabberImageBW8 class is used to wrap an EGrabber buffer whose pixel type is "Mono8". It can be used in Open eVision processing as an EImageBW8 object.

Base Class: [EImageBW8](#)

Namespace: Euresys.Open_eVision_2_10.EGrabberBridge

Methods

EGrabberImageBW8

E Constructs an EGrabberImageBW8.
G

GrabberImageBW8.EGrabberImageBW8

Constructs an EGrabberImageBW8.

Namespace: Euresys.Open_eVision_2_10.EGrabberBridge

```
[C#]
void EGrabberImageBW8 (
    BufferInfo infos
)
void EGrabberImageBW8 (
    ref FormatConvert converter,
    BufferInfo infos
)
void EGrabberImageBW8 (
    Euresys.Open_eVision_2_10.EGrabberBridge.EGrabberImageBW8
)
```

Parameters

infos

Information pertaining to an EGrabber buffer

converter

EGrabber format converter

-

Remarks

The FormatConverter class allows automatic pixel format conversions. If no format converter is provided, the pixel format of the buffer must be "Mono8"

4.77. EGrabberImageC24 Class

The EGrabberImageC24 class is used to wrap an EGrabber buffer whose pixel type is "BGR8". It can be used in Open eVision processing as an EImageC24 object.

Base Class: [EImageC24](#)

Namespace: Euresys.Open_eVision_2_10.EGrabberBridge

Methods

[EGrabberImageC24](#)

E Constructs an EGrabberImageC24.

G

GrabberImageC24.EGrabberImageC24

Constructs an EGrabberImageC24.

Namespace: Euresys.Open_eVision_2_10.EGrabberBridge


```
[C#]
void EGrabberImageC24 (
    BufferInfo infos
)

void EGrabberImageC24 (
    ref FormatConvert converter,
    BufferInfo infos
)

void EGrabberImageC24 (
    Euresys.Open_eVision_2_10.EGrabberBridge.EGrabberImageC24
)
```

Parameters

infos

Information pertaining to an EGrabber buffer

converter

EGrabber format converter

-

Remarks

The FormatConverter class allows automatic pixel format conversions. If no format converter is provided, the pixel format of the buffer must be "BGR8"

4.78. EGrayscaleDoubleThresholdSegmenter Class

Segments an image using a double threshold on a grayscale image.

Remarks

This segmenter is applicable to [EROIBW8](#) and [EROIBW16](#) grayscale images. It produces coded images with three layers: The Black layer (usually, with index 0) contains the unmasked pixels having a gray value strictly below the low threshold value; the White layer (usually, with index 2) contains the unmasked pixels having a gray value above or equal to the high threshold value; and the Neutral layer (usually, with index 1) contains the remaining unmasked pixels.

Base Class: [EThreeLayersImageSegmenter](#)

Namespace: Euresys.Open_eVision_2_10.Segmenters

Properties

HighThreshold

Value of the high threshold.

LowThreshold

Value of the low threshold.

G

rayscaleDoubleThresholdSegmenter.HighThreshold

Value of the high threshold.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]
```

```
uint HighThreshold
```

```
{ get; set; }
```

EGrayscaleDoubleThresholdSegmenter.LowThreshold

Value of the low threshold.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]
```

```
uint LowThreshold
```

```
{ get; set; }
```

4.79. EGrayscaleSingleThresholdSegmenter Class

Segments an image using a single threshold on a grayscale image.

Remarks

This segmenter is applicable to [EROIBW8](#) and [EROIBW16](#) grayscale images. It produces coded images with two layers: the black layer (usually, with index 0) contains the unmasked pixels having a gray value strictly below the threshold value; and the white layer (usually, with index 1) contains the remaining unmasked pixels, i.e. unmasked pixels having a gray value greater or equal to the threshold value. The default thresholding method is minimum residue. If another method is required, the [EGrayscaleSingleThresholdSegmenter::Mode](#) property must be set prior to encoding.

Base Class: [ETwoLayersImageSegmenter](#)

Namespace: `Euresys.Open_eVision_2_10.Segmenters`

Properties

AbsoluteThreshold

Value of the threshold to be used in the case of absolute thresholding.

LastThreshold

Retrieves the actual threshold value that was used for the last image that got encoded by the segmenter.

Mode

Threshold selection mode.

RelativeThreshold

Fraction of the image pixels that belongs to the black layer in the case of relative thresholding.

Methods

IsFirstApplication

E Checks whether the segmenter has already been used to segment an image.

G

rayscaleSingleThresholdSegmenter.AbsoluteThreshold

Value of the threshold to be used in the case of absolute thresholding.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]  
uint AbsoluteThreshold  
  
{ get; set; }
```

EGrayscaleSingleThresholdSegmenter.IsFirstApplication

Checks whether the segmenter has already been used to segment an image.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]  
bool IsFirstApplication(  
    )
```

EGrayscaleSingleThresholdSegmenter.LastThreshold

Retrieves the actual threshold value that was used for the last image that got encoded by the segmenter.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]  
uint LastThreshold  
  
    { get; }
```

Remarks

A call to this method will result in an exception if it is the first time the segmenter is applied. To check whether the segmenter has already been applied, call the [EGrayscaleSingleThresholdSegmenter::IsFirstApplication](#) method.

EGrayscaleSingleThresholdSegmenter.Mode

Threshold selection mode.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]  
Euresys.Open_eVision_2_10.EGrayscaleSingleThreshold Mode  
  
    { get; set; }
```

EGrayscaleSingleThresholdSegmenter.RelativeThreshold

Fraction of the image pixels that belongs to the black layer in the case of relative thresholding.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]
```

```
float RelativeThreshold
```

```
{ get; set; }
```

4.80. EHarrisCornerDetector Class

Manages a complete context for the Harris corner detector.

Remarks

This implementation of the Harris corner detector operates exclusively on a grayscale BW8 images.

Namespace: Euresys.Open_eVision_2_10

Properties

DerivationScale

Sets the derivation scale.

GradientNormalizationEnabled

Sets whether the gradient is normalized before the computation of the cornerness measure.

IntegrationScale

The integration scale.

SubpixelPrecisionEnabled

Sets whether the sub-pixel interpolation is enabled.

Threshold

Threshold on the cornerness measure for a pixel to be considered as a corner.

ThresholdingMode

M Thresholding mode for the cornerness measure.

e

Methods

Apply

Apply the Harris corner detector on an image/ROI.

EHarrisCornerDetector

EHarrisCornerDetector.Apply

Constructs an `EHarrisCornerDetector` object initialized to its default

Apply the Harris corner detector on an image/ROI.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void Apply(  
    Euresys.Open_eVision_2_10.EROIBW8 source,  
    Euresys.Open_eVision_2_10.EHarrisInterestPoints interestPoints  
)
```

Parameters

source

The source image/ROI.

interestPoints

The container in which to store the interest points.

EHarrisCornerDetector.DerivationScale

Sets the derivation scale.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float DerivationScale
{ get; set; }
```

Remarks

The derivation scale is the standard deviation of the Gaussian Filter used for the noise reduction during the computation of the gradient. Whenever the integration scale is set through [EHarrisCornerDetector::IntegrationScale](#), the derivation scale is reset to its default value, **0.7 * integrationScale**. This is a recommended value, as suggested by the literature.

EHarrisCornerDetector.EHarrisCornerDetector

Constructs a EHarrisCornerDetector object initialized to its default values.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EHarrisCornerDetector (
    Euresys.Open_eVision_2_10.EHarrisCornerDetector other
)
void EHarrisCornerDetector (
)
```

Parameters

other

-

EHarrisCornerDetector.GradientNormalizationEnabled

Sets whether the gradient is normalized before the computation of the cornerness measure.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool GradientNormalizationEnabled  
  
{ get; set; }
```

Remarks

If this flag is enabled, the values of the X-gradient and of the Y-gradient are first divided by their maximum absolute value (in the internal computations). This results in a cornerness measure that is roughly distributed around the value 1. If this flag is disabled, the cornerness measure will be much greater.

EHarrisCornerDetector.IntegrationScale

The integration scale.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float IntegrationScale  
  
{ get; set; }
```

Remarks

The *integration scale* is the standard deviation of the Gaussian filter that is used for scale analysis.

EHarrisCornerDetector.SubpixelPrecisionEnabled

Sets whether the sub-pixel interpolation is enabled.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
bool SubpixelPrecisionEnabled
```

```
{ get; set; }
```

Remarks

When this flag is enabled, a sub-pixel interpolation is carried on so as to improve the accuracy of the location of the corners, to the expense of a loss of speed.

EHarrisCornerDetector.Threshold

Threshold on the cornerness measure for a pixel to be considered as a corner.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float Threshold
```

```
{ get; set; }
```

Remarks

If the threshold mode is set to [Absolute](#), the threshold value is interpreted as an absolute threshold on the cornerness. In this case, the threshold must be a strictly positive real value.

If the threshold mode is set to [Relative](#), the threshold is expressed as a fraction ranging from 0 to 1 of the maximum value of the cornerness of the source image.

EHarrisCornerDetector.ThresholdingMode

Thresholding mode for the cornerness measure.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EHarrisThresholdingMode ThresholdingMode  
  
{ get; set; }
```

4.81. EHarrisInterestPoints Class

Container class for the results of the Harris corner detector.

Remarks

The [EHarrisCornerDetector](#) class stores its results in this container.

Namespace: Euresys.Open_eVision_2_10

Properties

PointCount

M The number of corner points in the container.

e

Methods

Draw

Draws the location of the corner points.

DrawCorner

Draws the location of a specific corner point.

DrawCornerWithCurrentPen

Draws the location of a specific corner point.

DrawWithCurrentPen

Draws the location of the corner points.

[EHarrisInterestPoints](#)

Constructs a container for the results of a Harris corner detector.

[GetCorners](#)

Returns the corners measure of a corner point.

[GetGradientMagnitude](#)

Returns the magnitude of the gradient at a corner point.

[GetGradientOrientation](#)

Returns the orientation of the gradient at a corner point.

[GetGradientX](#)

Returns the gradient along the X-axis of a corner point.

[GetGradientY](#)

Returns the gradient along the Y-axis of a corner point.

[GetPoint](#)

Returns the coordinates of a corner point.

[GetX](#)

Returns the abscissa of a corner point.

[GetY](#)

Returns the ordinate of a corner point.

H

arrisInterestPoints.Draw

Draws the location of the corner points.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void Draw(  
    IntPtr graphicContext,  
    float zoomX,  
    float zoomY,  
    float originX,  
    float originY  
)  
  
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ERGBColor color,  
    float zoomX,  
    float zoomY,  
    float originX,  
    float originY  
)  
  
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    float zoomX,  
    float zoomY,  
    float originX,  
    float originY  
)
```

Parameters

graphicContext

Graphic context on which to draw.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

originX

Horizontal panning factor. By default, no panning occurs.

originY

Vertical panning factor. By default, no panning occurs.

color

The color in which to draw the overlay.

Remarks

Drawing is done in the device context associated to the desired window.

EHarrisInterestPoints.DrawCorner

Draws the location of a specific corner point.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void DrawCorner(
    IntPtr graphicContext,
    int index,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)

void DrawCorner(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    int index,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)

void DrawCorner(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    int index,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)
```

Parameters

graphicContext

Graphic context on which to draw.

index

Corner index

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

originX

Horizontal panning factor. By default, no panning occurs.

originY

Vertical panning factor. By default, no panning occurs.

color

The color in which to draw the overlay.

Remarks

Drawing is done in the device context associated to the desired window.

EHarrisInterestPoints.DrawCornerWithCurrentPen

Draws the location of a specific corner point.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawCornerWithCurrentPen(
    IntPtr graphicContext,
    int index,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)
```

Parameters

graphicContext

Graphic context on which to draw.

index

Corner index

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

originX

Horizontal panning factor. By default, no panning occurs.

originY

Vertical panning factor. By default, no panning occurs.

Remarks

Drawing is done in the device context associated to the desired window.

EHarrisInterestPoints.DrawWithCurrentPen

Draws the location of the corner points.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawWithCurrentPen(
    IntPtr graphicContext,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)
```

Parameters

graphicContext

Graphic context on which to draw.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

originX

Horizontal panning factor. By default, no panning occurs.

originY

Vertical panning factor. By default, no panning occurs.

Remarks

Drawing is done in the device context associated to the desired window.

EHarrisInterestPoints.EHarrisInterestPoints

Constructs a container for the results of a Harris corner detector.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EHarrisInterestPoints (
    Euresys.Open_eVision_2_10.EHarrisInterestPoints other
)
void EHarrisInterestPoints (
)
```

Parameters

other

-

EHarrisInterestPoints.GetCornersness

Returns the cornersness measure of a corner point.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float GetCornersness (
    uint index
)
```

Parameters

index

The index of the corner point.

EHarrisInterestPoints.GetGradientMagnitude

Returns the magnitude of the gradient at a corner point.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float GetGradientMagnitude (
    uint index
)
```

Parameters

index

The index of the corner point.

EHarrisInterestPoints.GetGradientOrientation

Returns the orientation of the gradient at a corner point.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float GetGradientOrientation(
    uint index
)
```

Parameters

index

The index of the corner point.

EHarrisInterestPoints.GetGradientX

Returns the gradient along the X-axis of a corner point.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float GetGradientX(
    uint index
)
```

Parameters

index

The index of the corner point.

EHarrisInterestPoints.GetGradientY

Returns the gradient along the Y-axis of a corner point.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float GetGradientY(
    uint index
)
```

Parameters

index

The index of the corner point.

EHarrisInterestPoints.GetPoint

Returns the coordinates of a corner point.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint GetPoint(
    uint index
)
```

Parameters

index

The index of the corner point.

EHarrisInterestPoints.GetX

Returns the abscissa of a corner point.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float GetX(
    uint index
)
```

Parameters

index

The index of the corner point.

EHarrisInterestPoints.GetY

Returns the ordinate of a corner point.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float GetY(
    uint index
)
```

Parameters

index

The index of the corner point.

EHarrisInterestPoints.PointCount

The number of corner points in the container.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint PointCount
```

```
{ get; }
```

4.82. EHDRColorFuser Class

A [EHDRColorFuser](#) instance is a tool that flexibly fuses color images using HDR principles.

Namespace: Euresys.Open_eVision_2_10

Methods

EHDRColorFuser

Constructors for EHDRColorFuser objects. The image used must be the lightest (slowest shutter speed).

Fuse

Fuses a dark image with the light image passed during object construction.

GetFusedImage

Retrieves the resulting image. Returns false if part of the potential image dynamic is lost due to the chosen output image type.

operator=

Assignment operator

EHDRColorFuser.EHDRColorFuser

Constructors for EHDRColorFuser objects. The image used must be the lightest (slowest shutter speed).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EHDRColorFuser(
    Euresys.Open_eVision_2_10.EROIC24 lightSrc
)
void EHDRColorFuser(
    Euresys.Open_eVision_2_10.EROIC48 lightSrc
)
void EHDRColorFuser(
    Euresys.Open_eVision_2_10.EHDRColorFuser other
)
```

Parameters

lightSrc

-

other

-

EHDRColorFuser.Fuse

Fuses a dark image with the light image passed during object construction.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void Fuse(  
    Euresys.Open_eVision_2_10.EROIC24 darkSrc,  
    int shutterSpeedFactor  
)  
  
void Fuse(  
    Euresys.Open_eVision_2_10.EROIC48 darkSrc,  
    int shutterSpeedFactor  
)
```

Parameters

darkSrc

Dark input image (higher shutter speed).

shutterSpeedFactor

Shutter speed factor between light and dark image.

EHDRColorFuser.GetFusedImage

Retrieves the resulting image. Returns false if part of the potential image dynamic is lost due to the chosen output image type.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
bool GetFusedImage(  
    Euresys.Open_eVision_2_10.EROIC24 dst  
)  
  
bool GetFusedImage(  
    Euresys.Open_eVision_2_10.EROIC48 dst  
)
```

Parameters

dst

Output image.

EHDRColorFuser.operator=

Assignment operator

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EHDRColorFuser operator=(
    Euresys.Open_eVision_2_10.EHDRColorFuser other
)
```

Parameters

other

-

4.83. EHDRFuser Class

A [EHDRFuser](#) instance is a tool that flexibly fuses grayscale images using HDR principles.

Namespace: Euresys.Open_eVision_2_10

Methods

[EHDRFuser](#)

Constructors for EHDRFuser objects. The image used must be the lightest (slowest shutter speed).

[Fuse](#)

Fuses a dark image with the light image passed during object construction.

GetFusedImage

Retrieves the resulting image. Returns false if part of the potential image dynamic is lost due to the chosen output image type.

operator=

Assignment operator

H

DRFuser.EHDRFuser

Constructors for EHDRFuser objects. The image used must be the lightest (slowest shutter speed).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EHDRFuser (
    Euresys.Open_eVision_2_10.EROIBW8 lightSrc
)
void EHDRFuser (
    Euresys.Open_eVision_2_10.EROIBW16 lightSrc
)
void EHDRFuser (
    Euresys.Open_eVision_2_10.EHDRFuser other
)
```

Parameters

lightSrc

-

other

-

EHDRFuser.Fuse

Fuses a dark image with the light image passed during object construction.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Fuse(
    Euresys.Open_eVision_2_10.EROIBW8 darkSrc,
    int shutterSpeedFactor
)
void Fuse(
    Euresys.Open_eVision_2_10.EROIBW16 darkSrc,
    int shutterSpeedFactor
)
```

Parameters

darkSrc

Dark input image (higher shutter speed).

shutterSpeedFactor

Shutter speed factor between light and dark image.

EHDRFuser.GetFusedImage

Retrieves the resulting image. Returns false if part of the potential image dynamic is lost due to the chosen output image type.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
bool GetFusedImage(  
    Euresys.Open_eVision_2_10.EROIBW8 dst  
)  
  
bool GetFusedImage(  
    Euresys.Open_eVision_2_10.EROIBW16 dst  
)  
  
bool GetFusedImage(  
    Euresys.Open_eVision_2_10.EROIBW32 dst  
)
```

Parameters

dst

Output image.

EHDRFuser.operator=

Assignment operator

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EHDRFuser operator=(  
    Euresys.Open_eVision_2_10.EHDRFuser other  
)
```

Parameters

other

-

4.84. EHitAndMissKernel Class

Class that defines a kernel for the morphological hit-and-miss operations.

Namespace: Euresys.Open_eVision_2_10

Properties

EndX	Returns the abscissa of the rightmost element of the kernel.
EndY	Returns the abscissa of the bottommost element of the kernel.
StartX	Returns the abscissa of the leftmost element of the kernel.
StartY	M Returns the ordinate of the toptmost element of the kernel.

Methods

EHitAndMissKernel	Constructs a EHitAndMissKernel object.
GetValue	Returns the value of an element of the kernel at a given coordinate.
SetSize	Modify the size of the kernel.

SetValue

⌊ Sets the value of an element of the kernel at a given coordinate.

H

EHHitAndMissKernel.EHitAndMissKernel

Constructs a EHHitAndMissKernel object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EHHitAndMissKernel (
    Euresys.Open_eVision_2_10.EHitAndMissKernel other
)

void EHHitAndMissKernel (
    int startX,
    int startY,
    int endX,
    int endY
)

void EHHitAndMissKernel (
    uint halfSizeX,
    uint halfSizeY
)

void EHHitAndMissKernel (
)
```

Parameters

other

-

startX

The abscissa of the top leftmost element of the kernel. This value must be less than or equal to zero.

startY

The ordinate of the top leftmost element of the kernel. This value must be less than or equal to zero.

endX

The abscissa of the bottom rightmost element of the kernel. This value must be greater than or equal to zero.

endY

The ordinate of the bottom rightmost element of the kernel. This value must be greater than or equal to zero.

halfSizeX

The half of the kernel width minus 1. This value must be greater than zero.

halfSizeY

The half of the kernel height minus 1. This value must be greater than zero.

Remarks

The constructor without argument creates a centered kernel of size 3x3.

All the elements of the kernel are initialized with [DontCare](#) values.

If the object is constructed by specifying the halves of the kernel dimensions, the width (resp. the height) of the kernel is given by "2 * halfSizeX + 1" (resp. "2 * halfSizeY + 1"). Otherwise, the width (resp. the height) of the kernel is given by "endX - startX + 1" (resp. "endY - startY + 1").

EHitAndMissKernel.EndX

Returns the abscissa of the rightmost element of the kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int EndX
{ get; }
```

EHitAndMissKernel.EndY

Returns the abscissa of the bottommost element of the kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int EndY
{ get; }
```

EHitAndMissKernel.GetValue

Returns the value of an element of the kernel at a given coordinate.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EHitAndMissValue GetValue(
    int x,
    int y
)
```

Parameters

- x*
The abscissa of the element.
- y*
The ordinate of the element.

EHitAndMissKernel.SetSize

Modify the size of the kernel.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
void SetSize(
    int startX,
    int startY,
    int endX,
    int endY
)

void SetSize(
    uint halfSizeX,
    uint halfSizeY
)
```

Parameters

startX

The abscissa of the top leftmost element of the kernel. This value must be less than or equal to zero.

startY

The ordinate of the top leftmost element of the kernel. This value must be less than or equal to zero.

endX

The abscissa of the bottom rightmost element of the kernel. This value must be greater than or equal to zero.

endY

The ordinate of the bottom rightmost element of the kernel. This value must be greater than or equal to zero.

halfSizeX

The half of the kernel width minus 1. This value must be greater than zero.

halfSizeY

The half of the kernel height minus 1. This value must be greater than zero.

Remarks

All the elements of the kernel are initialized with [DontCare](#) values.

If the object is constructed by specifying the halves of the kernel dimensions, the width (resp. the height) of the kernel is given by "2 * halfSizeX + 1" (resp. "2 * halfSizeY + 1"). Otherwise, the width (resp. the height) of the kernel is given by "endX - startX + 1" (resp. "endY - startY + 1").

EHitAndMissKernel.SetValue

Sets the value of an element of the kernel at a given coordinate.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void SetValue(  
    int x,  
    int y,  
    Euresys.Open_eVision_2_10.EHitAndMissValue value  
)
```

Parameters

- x*
The abscissa of the element.
- y*
The ordinate of the element.
- value*
The value of the element.

EHitAndMissKernel.StartX

Returns the abscissa of the leftmost element of the kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
int StartX  
  
    { get; }
```

EHitAndMissKernel.StartY

Returns the ordinate of the toptmost element of the kernel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int StartY  
    { get; }
```

4.85. EHole Class

This class represents a hole inside an object (blob) of an encoded image.

Remarks

This class inherits from the ECodedElement class and provides an additional method to retrieve the parent object of a particular hole.

Base Class: [ECodedElement](#)

Namespace: Euresys.Open_eVision_2_10

Properties

[ParentObjectIndex](#)

┌ Returns the index of the parent object of the hole.
└
H

ole.ParentObjectIndex

Returns the index of the parent object of the hole.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint ParentObjectIndex
```

```
{ get; }
```

4.86. ElmageBW1 Class

The ElmageBW1 class is used to represent rectangular regions of interest inside [EBW1](#) black and white images. See ROIs.

Base Class: [EROIBW1](#)

Namespace: Euresys.Open_eVision_2_10

Methods

[ElmageBW1](#)

Constructs a ElmageBW1 image.

[GetBitIndex](#)

-

[operator=](#)

⌊ Copies a ElmageBW1 image.

ElmageBW1.ElmageBW1

Constructs a ElmageBW1 image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EImageBW1 (
)

void EImageBW1 (
    int width,
    int height
)

void EImageBW1 (
    Euresys.Open_eVision_2_10.EImageBW1 other
)
```

Parameters

width

The width, in pixels.

height

The height, in pixels.

other

Another EImageBW1 object.

Remarks

The constructor with no arguments creates a zero-sized image. You can modify the image size by calling [EBaseROI::SetSize](#) method. The sizing constructor constructs an image of the given size. See [EBaseROI::SetSize](#) for informations about the sizing restrictions. The copy constructor copies all the supplied image properties (content, attributes, sub-ROIs...) into the current object.

EImageBW1.GetBitIndex

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
System.UInt64 GetBitIndex(  
    int x,  
    int y  
)
```

Parameters

x

-

y

-

ElmageBW1.operator=

Copies a ElmageBW1 image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EImageBW1 operator=(  
    Euresys.Open_eVision_2_10.EImageBW1 other  
)
```

Parameters

other

Another ElmageBW1 object.

Remarks

This method copies all the supplied image properties (content, attributes, sub-ROIs...) into the current object.

4.87. EImageBW16 Class

The EImageBW16 class is used to represent rectangular regions of interest inside [EBW16](#) gray-level images. See ROIs.

Base Class: [EROIBW16](#)

Derived Class(es): [EGrabberImageBW16](#)

Namespace: Euresys.Open_eVision_2_10

Methods

[EImageBW16](#)

Constructs a EImageBW16 image.

[operator=](#)

Copies a EImageBW16 image.

imageBW16.EImageBW16

Constructs a EImageBW16 image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EImageBW16 (
)
void EImageBW16 (
    int width,
    int height
)
```

```
void EImageBW16(  
    Euresys.Open_eVision_2_10.EImageBW16 other  
)
```

Parameters

width

The width, in pixels.

height

The height, in pixels.

other

Another EImageBW16 object.

Remarks

The constructor with no arguments creates a zero-sized image. You can modify the image size by calling [EBaseROI::SetSize](#) method. The sizing constructor constructs an image of the given size. See [EBaseROI::SetSize](#) for informations about the sizing restrictions. The copy constructor copies all the supplied image properties (content, attributes, sub-ROIs...) into the current object.

EImageBW16.operator=

Copies a EImageBW16 image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EImageBW16 operator=(  
    Euresys.Open_eVision_2_10.EImageBW16 other  
)
```

Parameters

other

Another EImageBW16 object.

Remarks

This method copies all the supplied image properties (content, attributes, sub-ROIs...) into the current object.

4.88. EImageBW32 Class

The EImageBW32 class is used to represent rectangular regions of interest inside [EBW32](#) gray-level images. See ROIs.

Base Class: [EROIBW32](#)

Namespace: Euresys.Open_eVision_2_10

Methods

[EImageBW32](#)

Constructs a EImageBW32 image.

[operator=](#)

Copies a EImageBW32 image.

imageBW32.EImageBW32

Constructs a EImageBW32 image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EImageBW32 (
)
void EImageBW32 (
    int width,
    int height
)
```

```
void EImageBW32(  
    Euresys.Open_eVision_2_10.EImageBW32 other  
)
```

Parameters

width

The width, in pixels.

height

The height, in pixels.

other

Another EImageBW32 object.

Remarks

The constructor with no arguments creates a zero-sized image. You can modify the image size by calling [EBaseROI::SetSize](#) method. The sizing constructor constructs an image of the given size. See [EBaseROI::SetSize](#) for informations about the sizing restrictions. The copy constructor copies all the supplied image properties (content, attributes, sub-ROIs...) into the current object.

EImageBW32.operator=

Copies a EImageBW32 image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EImageBW32 operator=(  
    Euresys.Open_eVision_2_10.EImageBW32 other  
)
```

Parameters

other

Another EImageBW32 object.

Remarks

This method copies all the supplied image properties (content, attributes, sub-ROIs...) into the current object.

4.89. EImageBW8 Class

The EImageBW8 class is used to represent rectangular regions of interest inside [EBW8](#) gray-level images. See ROIs.

Base Class: [EROIBW8](#)

Derived Class(es): [EGrabberImageBW8](#)

Namespace: Euresys.Open_eVision_2_10

Methods

[EImageBW8](#)

Constructs a EImageBW8 image.

[operator=](#)

Copies a EImageBW8 image.

[EImageBW8.EImageBW8](#)

Constructs a EImageBW8 image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EImageBW8 (
)
void EImageBW8 (
    int width,
    int height
)
```

```
void EImageBW8 (
    Euresys.Open_eVision_2_10.EImageBW8 other
)
```

Parameters

width

The width, in pixels.

height

The height, in pixels.

other

Another EImageBW8 object.

Remarks

The constructor with no arguments creates a zero-sized image. You can modify the image size by calling [EBaseROI::SetSize](#) method. The sizing constructor constructs an image of the given size. See [EBaseROI::SetSize](#) for informations about the sizing restrictions. The copy constructor copies all the supplied image properties (content, attributes, sub-ROIs...) into the current object.

EImageBW8.operator=

Copies a EImageBW8 image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EImageBW8 operator=(
    Euresys.Open_eVision_2_10.EImageBW8 other
)
```

Parameters

other

Another EImageBW8 object.

Remarks

This method copies all the supplied image properties (content, attributes, sub-ROIs...) into the current object.

4.90. EImageC15 Class

The EImageC15 class is used to represent rectangular regions of interest inside [EC15](#) color images. See ROIs.

Base Class: [EROIC15](#)

Namespace: Euresys.Open_eVision_2_10

Methods

[EImageC15](#)

Constructs a EImageC15 image.

[operator=](#)

Copies a EImageC15 image.

imageC15.EImageC15

Constructs a EImageC15 image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EImageC15 (
)
void EImageC15 (
    int width,
    int height
)
```

```
void EImageC15 (
    Euresys.Open_eVision_2_10.EImageC15 other
)
```

Parameters

width

The width, in pixels.

height

The height, in pixels.

other

Another EImageC15 object.

Remarks

The constructor with no arguments creates a zero-sized image. You can modify the image size by calling [EBaseROI::SetSize](#) method. The sizing constructor constructs an image of the given size. See [EBaseROI::SetSize](#) for informations about the sizing restrictions. The copy constructor copies all the supplied image properties (content, attributes, sub-ROIs...) into the current object.

EImageC15.operator=

Copies a EImageC15 image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EImageC15 operator=(
    Euresys.Open_eVision_2_10.EImageC15 other
)
```

Parameters

other

Another EImageC15 object.

Remarks

This method copies all the supplied image properties (content, attributes, sub-ROIs...) into the current object.

4.91. EImageC16 Class

The EImageC16 class is used to represent rectangular regions of interest inside [EC16](#) color images. See ROIs.

Base Class: [EROIC16](#)

Namespace: Euresys.Open_eVision_2_10

Methods

[EImageC16](#)

Constructs a EImageC16 image.

[operator=](#)

Copies a EImageC16 image.

imageC16.EImageC16

Constructs a EImageC16 image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EImageC16 (
)
void EImageC16 (
    int width,
    int height
)
```

```
void EImageC16 (
    Euresys.Open_eVision_2_10.EImageC16 other
)
```

Parameters

width

The width, in pixels.

height

The height, in pixels.

other

Another EImageC16 object.

Remarks

The constructor with no arguments creates a zero-sized image. You can modify the image size by calling [EBaseROI::SetSize](#) method. The sizing constructor constructs an image of the given size. See [EBaseROI::SetSize](#) for informations about the sizing restrictions. The copy constructor copies all the supplied image properties (content, attributes, sub-ROIs...) into the current object.

EImageC16.operator=

Copies a EImageC16 image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EImageC16 operator=(
    Euresys.Open_eVision_2_10.EImageC16 other
)
```

Parameters

other

Another EImageC16 object.

Remarks

This method copies all the supplied image properties (content, attributes, sub-ROIs...) into the current object.

4.92. EImageC24 Class

The EImageC24 class is used to represent rectangular regions of interest inside [EC24](#) color images. See ROIs.

Base Class: [EROIC24](#)

Derived Class(es): [EGrabberImageC24](#)

Namespace: Euresys.Open_eVision_2_10

Methods

[EImageC24](#)

Constructs a EImageC24 image.

[operator=](#)

Copies a EImageC24 image.

imageC24.EImageC24

Constructs a EImageC24 image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EImageC24 (
)
void EImageC24 (
    int width,
    int height
)
```

```
void EImageC24 (  
    Euresys.Open_eVision_2_10.EImageC24 other  
)
```

Parameters

width

The width, in pixels.

height

The height, in pixels.

other

Another EImageC24 object.

Remarks

The constructor with no arguments creates a zero-sized image. You can modify the image size by calling [EBaseROI::SetSize](#) method. The sizing constructor constructs an image of the given size. See [EBaseROI::SetSize](#) for informations about the sizing restrictions. The copy constructor copies all the supplied image properties (content, attributes, sub-ROIs...) into the current object.

EImageC24.operator=

Copies a EImageC24 image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EImageC24 operator=(  
    Euresys.Open_eVision_2_10.EImageC24 other  
)
```

Parameters

other

Another EImageC24 object.

Remarks

This method copies all the supplied image properties (content, attributes, sub-ROIs...) into the current object.

4.93. EImageC24A Class

The EImageC24A class is used to represent rectangular regions of interest inside [EC24A](#) color images. See ROIs.

Base Class: [EROIC24A](#)

Namespace: Euresys.Open_eVision_2_10

Methods

[EImageC24A](#)

Constructs a EImageC24A image.

[operator=](#)

Copies a EImageC24A image.

imageC24A.EImageC24A

Constructs a EImageC24A image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EImageC24A(
)
void EImageC24A(
    int width,
    int height
)
```

```
void EImageC24A(  
    Euresys.Open_eVision_2_10.EImageC24A other  
)
```

Parameters

width

The width, in pixels.

height

The height, in pixels.

other

Another EImageC24A object.

Remarks

The constructor with no arguments creates a zero-sized image. You can modify the image size by calling [EBaseROI::SetSize](#) method. The sizing constructor constructs an image of the given size. See [EBaseROI::SetSize](#) for informations about the sizing restrictions. The copy constructor copies all the supplied image properties (content, attributes, sub-ROIs...) into the current object.

EImageC24A.operator=

Copies a EImageC24A image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EImageC24A operator=(  
    Euresys.Open_eVision_2_10.EImageC24A other  
)
```

Parameters

other

Another EImageC24A object.

Remarks

This method copies all the supplied image properties (content, attributes, sub-ROIs...) into the current object.

4.94. EImageC48 Class

The EImageC48 class is used to represent rectangular regions of interest inside EC48 color images. See ROIs.

Base Class: EROIC48

Namespace: Euresys.Open_eVision_2_10

Methods

EImageC48

Constructs a EImageC48 image.

operator=

Copies a EImageC48 image.

imageC48.EImageC48

Constructs a EImageC48 image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EImageC48 (
)
void EImageC48 (
    int width,
    int height
)
```

```
void EImageC48 (  
    Euresys.Open_eVision_2_10.EImageC48 other  
)
```

Parameters

width

The width, in pixels.

height

The height, in pixels.

other

Another EImageC48 object.

Remarks

The constructor with no arguments creates a zero-sized image. You can modify the image size by calling [EBaseROI::SetSize](#) method. The sizing constructor constructs an image of the given size. See [EBaseROI::SetSize](#) for informations about the sizing restrictions. The copy constructor copies all the supplied image properties (content, attributes, sub-ROIs...) into the current object.

EImageC48.operator=

Copies a EImageC48 image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EImageC48 operator=(  
    Euresys.Open_eVision_2_10.EImageC48 other  
)
```

Parameters

other

Another EImageC48 object.

Remarks

This method copies all the supplied image properties (content, attributes, sub-ROIs...) into the current object.

4.95. EImageEncoder Class

This class is responsible for the encoding of an image into an [ECodedImage2](#) object.

Remarks

This class is responsible for the extraction of the runs in a source image, the aggregation of the runs into objects, as well as the proper handling of the continuous mode. It also provides methods to configure the image segmentation process.

The segmentation process classifies the pixels of the source image according to their value to create a set of layers. In each layer taken separately, the encoding process then assembles the connected pixels to build the coded elements (blobs).

By default, the segmentation method consists of grayscale single thresholding, with automatic threshold selection (as determined by the minimum residue rule).

Namespace: Euresys.Open_eVision_2_10

Properties

BinaryImageSegmenter

Returns a reference to the internal instance of the the segmenter for binary images, in order to set its parameters.

ColorRangeThresholdSegmenter

Returns a reference to the internal instance of the the segmenter for color-range thresholding, in order to set its parameters.

ColorSingleThresholdSegmenter

Returns a reference to the internal instance of the the segmenter for single color thresholding, in order to set its parameters.

ContinuousModeEnabled

Continuous mode enabling status.

ContinuousModeMaxHeight

Maximum number of rows that are kept in memory in the continuous mode.

EncodingConnexity

Connexity mode.

GrayscaleDoubleThresholdSegmenter

Returns a reference to the internal instance of the the segmenter for double color thresholding, in order to set its parameters.

GrayscaleSingleThresholdSegmenter

Returns a reference to the internal instance of the the segmenter for single grayscale thresholding, in order to set its parameters.

ImageRangeSegmenter

Returns a reference to the internal instance of the the segmenter for pixel-by-pixel, range-based thresholding, in order to set its parameters.

LabeledImageSegmenter

Returns a reference to the internal instance of the the segmenter that maps the value of the pixels directly to a layer index, in order to set its parameters.

ReferenceImageSegmenter

Returns a reference to the internal instance of the the segmenter for single pixel-by-pixel thresholding, in order to set its parameters.

SegmentationMethod

M Segmentation method used during the encoding.

e

thods

EImageEncoder

Constructs an image encoder.

Encode

Encodes an image or an ROI as a coded image.

FlushContinuousMode

Resets the continuous mode, emptying the internal memory, after writing the objects that are not yet completed in a coded image.

ResetContinuousMode

Resets the continuous mode, emptying the internal memory.

ImageEncoder.BinaryImageSegmenter

Returns a reference to the internal instance of the the segmenter for binary images, in order to set its parameters.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.Segmenters.EBinaryImageSegmenter BinaryImageSegmenter  
{ get; }
```

ImageEncoder.ColorRangeThresholdSegmenter

Returns a reference to the internal instance of the the segmenter for color-range thresholding, in order to set its parameters.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.Segmenters.EColorRangeThresholdSegmenter  
ColorRangeThresholdSegmenter  
  
{ get; }
```

ElmageEncoder.ColorSingleThresholdSegmenter

Returns a reference to the internal instance of the the segmenter for single color thresholding, in order to set its parameters.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.Segmenters.EColorSingleThresholdSegmenter  
ColorSingleThresholdSegmenter  
  
{ get; }
```

ElmageEncoder.ContinuousModeEnabled

Continuous mode enabling status.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
bool ContinuousModeEnabled  
  
{ get; set; }
```

Remarks

In the continuous mode, objects are constructed over a sequence of images: The image encoder encodes only the objects that contain no run touching the last row of the source image. The objects touching the inferior border of the image are not written in the coded image: These objects are indeed expected to continue in the subsequent image chunks. Such objects are kept in memory, and are consumed when analyzing the subsequent images.

ElmageEncoder.ContinuousModeMaxHeight

Maximum number of rows that are kept in memory in the continuous mode.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint ContinuousModeMaxHeight
{ get; set; }
```

Remarks

This property can be used to put a bound on the size of the internal memory of the image encoder in the continuous mode. If this property is set to zero, then memory can grow arbitrarily (there is no maximum number of rows).

ElmageEncoder.EImageEncoder

Constructs an image encoder.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EImageEncoder(
    Euresys.Open_eVision_2_10.EImageEncoder other
)
```

```
void EImageEncoder(  
    )
```

Parameters

other

-

EImageEncoder.Encode

Encodes an image or an ROI as a coded image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void Encode(  
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,  
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage  
    )  
  
void Encode(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage  
    )  
  
void Encode(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage  
    )  
  
void Encode(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage  
    )
```

```
void Encode(  
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 inputMask,  
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage  
)  
  
void Encode(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 inputMask,  
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage  
)  
  
void Encode(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 inputMask,  
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage  
)  
  
void Encode(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 inputMask,  
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage  
)  
  
void Encode(  
    Euresys.Open_eVision_2_10.EROIBW1 sourceImage,  
    Euresys.Open_eVision_2_10.ERegion region,  
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage  
)  
  
void Encode(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.ERegion region,  
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage  
)  
  
void Encode(  
    Euresys.Open_eVision_2_10.EROIBW16 sourceImage,  
    Euresys.Open_eVision_2_10.ERegion region,  
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage  
)  
  
void Encode(  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.ERegion region,  
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage  
)
```

Parameters

sourceImage

The input image that is to be encoded.

codedImage

The coded image that will hold the result of the encoding process.

inputMask

The possible input Flexible Mask that restricts the encoding. The input mask is a grayscale image having the same height and the same width as the source image. Any pixel in the source image that is covered by a value of **0** in the input mask will not get encoded in any layer. Any other pixel value in the input mask causes the pixel to be a candidate for the encoding.

region

Region of the input image that is to be encoded.

Remarks

The previous content of the result coded image is discarded.

ElmageEncoder.EncodingConnexity

Connexity mode.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EEncodingConnexity EncodingConnexity
    { get; set; }
```

Remarks

The connexity mode specifies the conditions that must hold for neighboring pixels to belong to the same object.

ElmageEncoder.FlushContinuousMode

Resets the continuous mode, emptying the internal memory, after writing the objects that are not yet completed in a coded image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void FlushContinuousMode(  
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage  
)
```

Parameters

codedImage

The coded image in which the not-yet-completed objects are written.

ElmageEncoder.GrayscaleDoubleThresholdSegmenter

Returns a reference to the internal instance of the the segmenter for double color thresholding, in order to set its parameters.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
Euresys.Open_eVision_2_10.Segmenters.EGrayscaleDoubleThresholdSegmenter  
GrayscaleDoubleThresholdSegmenter  
  
{ get; }
```

ElmageEncoder.GrayscaleSingleThresholdSegmenter

Returns a reference to the internal instance of the the segmenter for single grayscale thresholding, in order to set its parameters.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.Segmenters.EGrayscaleSingleThresholdSegmenter  
GrayscaleSingleThresholdSegmenter  
  
{ get; }
```

ElmageEncoder.ImageRangeSegmenter

Returns a reference to the internal instance of the the segmenter for pixel-by-pixel, range-based thresholding, in order to set its parameters.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.Segmenters.EImageRangeSegmenter ImageRangeSegmenter  
  
{ get; }
```

ElmageEncoder.LabeledImageSegmenter

Returns a reference to the internal instance of the the segmenter thats maps the value of the pixels directly to a layer index, in order to set its parameters.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.Segmenters.ELabeledImageSegmenter LabeledImageSegmenter  
  
{ get; }
```


ElmageEncoder.ReferenceImageSegmenter

Returns a reference to the internal instance of the the segmenter for single pixel-by-pixel thresholding, in order to set its parameters.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.Segmenters.EReferenceImageSegmenter ReferenceImageSegmenter  
    { get; }
```

ElmageEncoder.ResetContinuousMode

Resets the continuous mode, emptying the internal memory.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void ResetContinuousMode(  
    )
```

ElmageEncoder.SegmentationMethod

Segmentation method used during the encoding.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.ESegmentationMethod SegmentationMethod  
{ get; set; }
```

4.96. EImageRangeSegmenter Class

Segments an image using a pixel-by-pixel double threshold given as two images.

Remarks

This segmenter is applicable to [EROIBW8](#), [EROIBW16](#) and [EROIC24](#) images. It produces coded images with two layers. The low threshold and the high threshold are defined for each pixel individually by means of two reference images of the same type as the source image: the Low Image and the High Image.

For grayscales images, the White layer (usually, with index 1) contains unmasked pixels having a gray value in a range defined by the gray value of the corresponding unmasked pixels in the Low Image and the High Image.

For RGB color images, the White layer (usually, with index 1) contains unmasked pixels having a color inside the cube of the color space defined by the colors of the corresponding unmasked pixels in the Low Image and the High Image.

The Black layer (usually, with index 0) contains the remaining unmasked pixels.

Base Class: [ETwoLayersImageSegmenter](#)

Namespace: Euresys.Open_eVision_2_10.Segmenters

Properties

[BlackLayerEncoded](#)

Black layer encoding status.

[BlackLayerIndex](#)

Index of the black layer in the destination coded image.

[HighImageBW16](#)

High image for the segmentation of [EROIBW16](#) images.

HighImageBW8

High image for the segmentation of [EROIBW8](#) images.

HighImageC24

High image for the segmentation of [EROIC24](#) images.

LowImageBW16

Low image for the segmentation of [EROIBW16](#) images.

LowImageBW8

Low image for the segmentation of [EROIBW8](#) images.

LowImageC24

Low image for the segmentation of [EROIC24](#) images.

WhiteLayerEncoded

White layer encoding status.

WhiteLayerIndex

Index of the white layer in the destination coded image.

imageRangeSegmenter.BlackLayerEncoded

Black layer encoding status.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]
```

```
override bool BlackLayerEncoded
```

```
{ get; set; }
```

ElmageRangeSegmenter.BlackLayerIndex

Index of the black layer in the destination coded image.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]  
override uint BlackLayerIndex  
  
    { get; set; }
```

Remarks

Setting this property automatically switches on the encoding of the black layer.

ElmageRangeSegmenter.HighImageBW16

High image for the segmentation of [EROIBW16](#) images.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]  
Euresys.Open_eVision_2_10.EROIBW16 HighImageBW16  
  
    { get; set; }
```

ElmageRangeSegmenter.HighImageBW8

High image for the segmentation of [EROIBW8](#) images.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]
```

```
Euresys.Open_eVision_2_10.EROIBW8 HighImageBW8
```

```
{ get; set; }
```

ElmageRangeSegmenter.HighImageC24

High image for the segmentation of [EROIC24](#) images.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]
```

```
Euresys.Open_eVision_2_10.EROIC24 HighImageC24
```

```
{ get; set; }
```

ElmageRangeSegmenter.LowImageBW16

Low image for the segmentation of [EROIBW16](#) images.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]
```

```
Euresys.Open_eVision_2_10.EROIBW16 LowImageBW16
```

```
{ get; set; }
```

ElmageRangeSegmenter.LowImageBW8

Low image for the segmentation of [EROIBW8](#) images.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]  
Euresys.Open_eVision_2_10.EROIBW8 LowImageBW8  
{ get; set; }
```

ElmageRangeSegmenter.LowImageC24

Low image for the segmentation of [EROIC24](#) images.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]  
Euresys.Open_eVision_2_10.EROIC24 LowImageC24  
{ get; set; }
```

ElmageRangeSegmenter.WhiteLayerEncoded

White layer encoding status.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]
override bool WhiteLayerEncoded
{ get; set; }
```

ElmageRangeSegmenter.WhiteLayerIndex

Index of the white layer in the destination coded image.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]
override uint WhiteLayerIndex
{ get; set; }
```

Remarks

Setting this property automatically switches on the encoding of the white layer.

4.97. ElmageSegmenter Class

Base class from which all the segmenters derive.

Derived Class(es): [ETwoLayersImageSegmenter](#) [EThreeLayersImageSegmenter](#) [ELabeledImageSegmenter](#)

Namespace: Euresys.Open_eVision_2_10.Segmenters

4.98. EIntegerRange Class

Represents a range of integer values.

Namespace: Euresys.Open_eVision_2_10

Properties

Center	Center of the range.
LowerBound	Lower bound of the range.
Size	Size of the range.
UpperBound	M Upper bound of the range.

Methods

EIntegerRange	Creates an EIntegerRange object.
IsInRange	Checks if a value is inside the range.
operator=	Assignment operator

[SetBounds](#)

Sets the bounds of the range.

[SetFromBaseAndAbsoluteTolerance](#)

Sets the bounds of the range from a base value and an absolute tolerance from this base value.

[SetFromBaseAndRelativeTolerance](#)

Sets the bounds of the range from a base value and a relative tolerance from this base value.

[Update](#)

Updates the range. If the value passed is outside of the range bounds, they are modified so they include the value.

IntegerRange.Center

Center of the range.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int Center
{ get; }
```

IntegerRange.IntegerRange

Creates an [IntegerRange](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EIntegerRange (
)

void EIntegerRange (
    int min,
    int max
)

void EIntegerRange (
    Euresys.Open_eVision_2_10.EIntegerRange range
)
```

Parameters

min

Lower bound of the range.

max

Upper bound of the range.

range

Range to copy.

EIntegerRange.IsInRange

Checks if a value is inside the range.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool IsInRange (
    int value,
    bool lowerBoundInclusive,
    bool upperBoundInclusive
)
```

```
bool IsInRange (  
    float value,  
    bool lowerBoundInclusive,  
    bool upperBoundInclusive  
)
```

Parameters

value

Value to test.

lowerBoundInclusive

Indicates if the lower bound should be considered inside the range (true) or outside (false).

upperBoundInclusive

Indicates if the upper bound should be considered inside the range (true) or outside (false).

IntegerRange.LowerBound

Lower bound of the range.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
int LowerBound  
{ get; }
```

IntegerRange.operator=

Assignment operator

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EIntegerRange operator=(
    Euresys.Open_eVision_2_10.EIntegerRange other
)
```

Parameters

other

-

EIntegerRange.SetBounds

Sets the bounds of the range.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetBounds (
    int min,
    int max
)
```

Parameters

min

Lower bound of the range.

max

Upper bound of the range.

EIntegerRange.SetFromBaseAndAbsoluteTolerance

Sets the bounds of the range from a base value and an absolute tolerance from this base value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void SetFromBaseAndAbsoluteTolerance (  
    int baseValue,  
    int tolerance  
)
```

Parameters

baseValue

Base value.

tolerance

Absolute tolerance around the base value. Must be positive.

Remarks

The range will be set with a lower bound of (base - tolerance) and an upper bound of (base + tolerance).

EIntegerRange.SetFromBaseAndRelativeTolerance

Sets the bounds of the range from a base value and a relative tolerance from this base value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void SetFromBaseAndRelativeTolerance (  
    int baseValue,  
    float tolerance  
)
```

Parameters

baseValue

Base value.

tolerance

Relative tolerance around the base value. Must be positive.

Remarks

The range will be set with a lower bound of $(base - (base * tolerance))$ and an upper bound of $(base + (base * tolerance))$.

IntegerRange.Size

Size of the range.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int Size  
{ get; }
```

IntegerRange.Update

Updates the range. If the value passed is outside of the range bounds, they are modified so they include the value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Update(  
    int value  
)
```

Parameters

value

Value to be included in the range.

EIntegerRange.UpperBound

Upper bound of the range.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int UpperBound  
    { get; }
```

4.99. EKernel Class

Kernel for use in convolution operations.

Namespace: Euresys.Open_eVision_2_10

Properties

Gain	Global gain.
Offset	Global offset (a constant added to the convolution result).
OutsideValue	Out-of-limits image value (only influences the result along image edges).
RawDataPtr	Pointer to the upper left convolution coefficient.

Rectifier

Rectification mode. This property allows specifying how negative convolution result values are handled.

SizeX

Number of coefficients along a row.

SizeY

M Number of coefficients along a column.

e

Methods

EKernel

Constructs an EKernel object.

GetKernelData

Returns the convolution coefficient of given indices.

SetKernelData

Sets the convolution coefficient values at the given indices.

SetSize

E⁻
K

ernel.EKernel

Constructs an EKernel object.

Namespace: Euresys.Open_eVision_2_10


```

[C#]
void EKernel(
    Euresys.Open_eVision_2_10.EKernel other
)

void EKernel(
)

void EKernel(
    short sizeX,
    short sizeY,
    float gain,
    uint offset,
    Euresys.Open_eVision_2_10.EKernelRectifier rectifier,
    uint outsideValue
)

void EKernel(
    Euresys.Open_eVision_2_10.EKernelType KernelType
)

```

Parameters

other

-

sizeX

Number of coefficients along a row.

sizeY

Number of coefficients along a column.

gain

Global gain.

offset

Global offset.

rectifier

Rectification mode, as defined by [EKernelRectifier](#).

outsideValue

Out-of-limits image value.

KernelType

Kernel type, as defined by [EKernelType](#).

Remarks

The default constructor constructs a void kernel. A void kernel has no associated convolution coefficients. The sizing constructor constructs a kernel of given size and global parameters. The third constructor constructs a kernel of a predefined type.

EKernel.Gain

Global gain.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
float Gain  
  
    { get; set; }
```

Remarks

Before the global gain is applied, the coefficients are normalized so that their sum equals one, unless their sum equals zero (as is the case for a derivation operator). The rectification enables to handle the negative values that may appear after convolution.

EKernel.GetKernelData

Returns the convolution coefficient of given indices.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void GetKernelData(  
    int columnIndex,  
    int rowIndex,  
    out float coefficientValue  
)
```

Parameters

columnIndex

Column index, from **0** on, increasing rightwards.

rowIndex

Row index, from **0** on, increasing downwards.

coefficientValue

Reference to the coefficient value.

EKernel.Offset

Global offset (a constant added to the convolution result).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
uint Offset  
  
{ get; set; }
```

EKernel.OutsideValue

Out-of-limits image value (only influences the result along image edges).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint OutsideValue
{ get; set; }
```

EKernel.RawDataPtr

Pointer to the upper left convolution coefficient.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
IntPtr RawDataPtr
{ get; }
```

Remarks

This pointer is actually the base address of a float array containing all coefficients.

EKernel.Rectifier

Rectification mode. This property allows specifying how negative convolution result values are handled.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EKernelRectifier Rectifier
{ get; set; }
```

EKernel.SetKernelData

Sets the convolution coefficient values at the given indices.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void SetKernelData(  
    int columnIndex,  
    int rowIndex,  
    float coefficientValue  
)
```

```
void SetKernelData(  
    float coefficientValue00,  
    float coefficientValue10,  
    float coefficientValue20,  
    float coefficientValue01,  
    float coefficientValue11,  
    float coefficientValue21,  
    float coefficientValue02,  
    float coefficientValue12,  
    float coefficientValue22  
)
```

```
void SetKernelData(  
    float coefficientValue00,  
    float coefficientValue10,  
    float coefficientValue20,  
    float coefficientValue30,  
    float coefficientValue40,  
    float coefficientValue01,  
    float coefficientValue11,  
    float coefficientValue21,  
    float coefficientValue31,  
    float coefficientValue41,  
    float coefficientValue02,  
    float coefficientValue12,  
    float coefficientValue22,  
    float coefficientValue32,  
    float coefficientValue42,  
    float coefficientValue03,  
    float coefficientValue13,  
    float coefficientValue23,  
    float coefficientValue33,  
    float coefficientValue43,  
    float coefficientValue04,  
    float coefficientValue14,  
    float coefficientValue24,  
    float coefficientValue34,  
    float coefficientValue44  
)
```

```
void SetKernelData(  
    float coefficientValue00,  
    float coefficientValue10,  
    float coefficientValue20,  
    float coefficientValue30,  
    float coefficientValue40,  
    float coefficientValue50,  
    float coefficientValue60,  
    float coefficientValue01,  
    float coefficientValue11,  
    float coefficientValue21,  
    float coefficientValue31,  
    float coefficientValue41,  
    float coefficientValue51,  
    float coefficientValue61,  
    float coefficientValue02,  
    float coefficientValue12,  
    float coefficientValue22,  
    float coefficientValue32,  
    float coefficientValue42,  
    float coefficientValue52,  
    float coefficientValue62,  
    float coefficientValue03,  
    float coefficientValue13,  
    float coefficientValue23,  
    float coefficientValue33,  
    float coefficientValue43,  
    float coefficientValue53,  
    float coefficientValue63,  
    float coefficientValue04,  
    float coefficientValue14,  
    float coefficientValue24,  
    float coefficientValue34,  
    float coefficientValue44,  
    float coefficientValue54,  
    float coefficientValue64,  
    float coefficientValue05,  
    float coefficientValue15,  
    float coefficientValue25,  
    float coefficientValue35,  
    float coefficientValue45,  
    float coefficientValue55,  
    float coefficientValue65,  
    float coefficientValue06,
```

```
float coefficientValue16,  
float coefficientValue26,  
float coefficientValue36,  
float coefficientValue46,  
float coefficientValue56,  
float coefficientValue66  
)
```

Parameters

columnIndex

Column index, from **0** on, increasing rightwards.

rowIndex

Row index, from **0** on, increasing downwards.

coefficientValue

New coefficientValue.

coefficientValue00

Coefficient value at corresponding column and row indices.

coefficientValue10

Coefficient value at corresponding column and row indices.

coefficientValue20

Coefficient value at corresponding column and row indices.

coefficientValue01

Coefficient value at corresponding column and row indices.

coefficientValue11

Coefficient value at corresponding column and row indices.

coefficientValue21

Coefficient value at corresponding column and row indices.

coefficientValue02

Coefficient value at corresponding column and row indices.

coefficientValue12

Coefficient value at corresponding column and row indices.

coefficientValue22

Coefficient value at corresponding column and row indices.

coefficientValue30

Coefficient value at corresponding column and row indices.

coefficientValue40

Coefficient value at corresponding column and row indices.

coefficientValue31

Coefficient value at corresponding column and row indices.

coefficientValue41

Coefficient value at corresponding column and row indices.

coefficientValue32

Coefficient value at corresponding column and row indices.

coefficientValue42

Coefficient value at corresponding column and row indices.

coefficientValue03

Coefficient value at corresponding column and row indices.

coefficientValue13

Coefficient value at corresponding column and row indices.

coefficientValue23

Coefficient value at corresponding column and row indices.

coefficientValue33

Coefficient value at corresponding column and row indices.

coefficientValue43

Coefficient value at corresponding column and row indices.

coefficientValue04

Coefficient value at corresponding column and row indices.

coefficientValue14

Coefficient value at corresponding column and row indices.

coefficientValue24

Coefficient value at corresponding column and row indices.

coefficientValue34

Coefficient value at corresponding column and row indices.

coefficientValue44

Coefficient value at corresponding column and row indices.

coefficientValue50

Coefficient value at corresponding column and row indices.

coefficientValue60

Coefficient value at corresponding column and row indices.

coefficientValue51

Coefficient value at corresponding column and row indices.

coefficientValue61

Coefficient value at corresponding column and row indices.

coefficientValue52

Coefficient value at corresponding column and row indices.

coefficientValue62

Coefficient value at corresponding column and row indices.

coefficientValue53

Coefficient value at corresponding column and row indices.

coefficientValue63

Coefficient value at corresponding column and row indices.

coefficientValue54

Coefficient value at corresponding column and row indices.

coefficientValue64

Coefficient value at corresponding column and row indices.

coefficientValue05

Coefficient value at corresponding column and row indices.

coefficientValue15

Coefficient value at corresponding column and row indices.

coefficientValue25

Coefficient value at corresponding column and row indices.

coefficientValue35

Coefficient value at corresponding column and row indices.

coefficientValue45

Coefficient value at corresponding column and row indices.

coefficientValue55

Coefficient value at corresponding column and row indices.

coefficientValue65

Coefficient value at corresponding column and row indices.

coefficientValue06

Coefficient value at corresponding column and row indices.

coefficientValue16

Coefficient value at corresponding column and row indices.

coefficientValue26

Coefficient value at corresponding column and row indices.

coefficientValue36

Coefficient value at corresponding column and row indices.

coefficientValue46

Coefficient value at corresponding column and row indices.

coefficientValue56

Coefficient value at corresponding column and row indices.

coefficientValue66

Coefficient value at corresponding column and row indices.

Remarks

The function can also set the coefficient values for 3x3, 5x5 and 7x7 kernels.

EKernel.SetSize

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void SetSize(  
    short n16SizeX,  
    short n16SizeY  
)
```

Parameters

n16SizeX

-

n16SizeY

-

EKernel.SizeX

Number of coefficients along a row.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
short SizeX
```

```
{ get; }
```

EKernel.SizeY

Number of coefficients along a column.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
short SizeY  
{ get; }
```

4.100. ELabeledImageSegmenter Class

Segments an image by mapping the value of the pixels directly to a layer index.

Remarks

This segmenter is applicable to [EROIBW8](#) and [EROIBW16](#) grayscale images. It produces coded images with a varying number of layers. The layer with index **N** contains all the unmasked pixels having a gray value equal to **N**.

By default, the segmentation is restricted to the range of layers whose index is between **0** and **255** (inclusive). This default range can be changed through [ELabeledImageSegmenter::MinLayer](#) and [ELabeledImageSegmenter::MaxLayer](#).

Base Class: [EImageSegmenter](#)

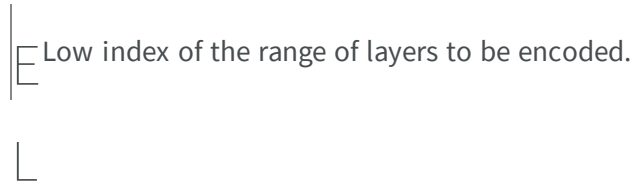
Namespace: Euresys.Open_eVision_2_10.Segmenters

Properties

[MaxLayer](#)

High index of the range of layers to be encoded.

MinLayer



ELabeledImageSegmenter.MaxLayer

High index of the range of layers to be encoded.

Namespace: Euresys.Open_eVision_2_10.Segmenters

[C#]

```
Euresys.Open_eVision_2_10.EBW16 MaxLayer  
{ get; set; }
```

ELabeledImageSegmenter.MinLayer

Low index of the range of layers to be encoded.

Namespace: Euresys.Open_eVision_2_10.Segmenters

[C#]

```
Euresys.Open_eVision_2_10.EBW16 MinLayer  
{ get; set; }
```

4.101. ELandmark Class

The landMark descriptor class.

Namespace: Euresys.Open_eVision_2_10

Properties

SensorX

SensorX.

SensorY

SensorY.

WorldX

WorldX.

WorldY

MWorldY.

e

Methods

ELandmark

Constructs a [ELandmark](#) object.

operator=

Assignment operator.

ELandmark.ELandmark

Constructs a [ELandmark](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ELandmark (
    Euresys.Open_eVision_2_10.ELandmark other
)
void ELandmark (
)
```

Parameters

other

-

ELandmark.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ELandmark operator=(
    Euresys.Open_eVision_2_10.ELandmark other
)
```

Parameters

other

[ELandmark](#) object to copy.

ELandmark.SensorX

SensorX.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float SensorX
```

```
{ get; set; }
```

ELandmark.SensorY

SensorY.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float SensorY
```

```
{ get; set; }
```

ELandmark.WorldX

WorldX.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
float WorldX
{ get; set; }
```

ELandmark.WorldY

WorldY.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float WorldY
{ get; set; }
```

4.102. ELaserLineExtractor Class

Manages a laser line extraction context.

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

[AnalysisMode](#)

Analysis mode.

[AnalysisThreshold](#)

Analysis threshold. Set this value to eliminate noise.

DepthMap

Returns the current depth map.

EnableSmoothing

Enables or disables grayscale profile smoothing before extraction.

Profile

M Returns the last extracted profile.

e

Methods

ELaserLineExtractor

Creates an [ELaserLineExtractor](#) object.

ExtractProfileFromFrame

Extracts a profile from a frame and adds it to the current depth map.
Returns true if the depth map is complete and ready for further processing.

operator=

Assignment operator

SetSmoothingParameters

E Sets the parameters of the smoothing kernel.

L

ELaserLineExtractor.AnalysisMode

Analysis mode.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EMaximumAnalysisMode AnalysisMode
{ get; set; }
```

ELaserLineExtractor.AnalysisThreshold

Analysis threshold. Set this value to eliminate noise.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
int AnalysisThreshold
{ get; set; }
```

Remarks

In the center of gravity (COG) analysis mode, this threshold is used to discriminate peaks and should be set accordingly.

ELaserLineExtractor.DepthMap

Returns the current depth map.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 DepthMap
{ get; }
```

Remarks

Should be called only when the previous call to `ExtractProfileFromFrame()` returned true. Otherwise, the depth map returns will be incomplete.

ELaserLineExtractor.ELaserLineExtractor

Creates an [ELaserLineExtractor](#) object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ELaserLineExtractor (
    int frameWidth,
    int frameHeight,
    int numFramesPerMap,
    float zResolution
)

void ELaserLineExtractor (
    Euresys.Open_eVision_2_10.Easy3D.ELaserLineExtractor other
)
```

Parameters

frameWidth

Width of the frames from which the profiles will be extracted.

frameHeight

Height of the frames from which the profiles will be extracted.

numFramesPerMap

Number of frames (and thus profiles) to be used per depth map. Each extracted profile create a line in the depth map.

zResolution

Optional parameter for the Z resolution of the extracted profile.

With a value of 0, the resolution will be automatically calculated to maximize the sub-pixel accuracy.

other

The object that should be copied

ELaserLineExtractor.EnableSmoothing

Enables or disables grayscale profile smoothing before extraction.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool EnableSmoothing
{ get; set; }
```

ELaserLineExtractor.ExtractProfileFromFrame

Extracts a profile from a frame and adds it to the current depth map.
Returns true if the depth map is complete and ready for further processing.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool ExtractProfileFromFrame (
    Euresys.Open_eVision_2_10.EROIBW8 frame
)
```

Parameters

frame

Frame from which the profile will be extracted.

ELaserLineExtractor.operator=

Assignment operator

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.Easy3D.ELaserLineExtractor operator=(  
    Euresys.Open_eVision_2_10.Easy3D.ELaserLineExtractor other  
)
```

Parameters

other

The object that should be copied

ELaserLineExtractor.Profile

Returns the last extracted profile.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float[] Profile  
  
    { get; }
```

Remarks

If a point could not be extracted, its value will be set to `FLOAT_MAX`.

ELaserLineExtractor.SetSmoothingParameters

Sets the parameters of the smoothing kernel.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetSmoothingParameters (
    int param0,
    int param1,
    int param2
)
```

Parameters

param0

First kernel parameter.

param1

Second kernel parameter.

param2

Third kernel parameter.

Remarks

If enabled, the smoothing will be performed using the following formula: $f[i] = (f[i-1] * param0) + (f[i] * param1) + (f[i+1] * param2)$.

4.103. ELine Class

Represents a model of a line segment in EasyGauge.

Base Class: [EFrame](#)

Namespace: Euresys.Open_eVision_2_10

Properties

End	End point coordinates of the ELine object.
Length	Length of the ELine object.
Org	M Origin point coordinates of the ELine object.

e

Methods

CopyTo	Copies all the data of the current ELine object into another ELine object and returns it.
ELine	Constructs a ELine object.
GetAngleBetweenLines	Computes the angle between two lines.
GetDistanceBetweenPointAndLine	Computes the distance between a point and a line.
GetIntersectionOfLines	Computes the intersection between two lines.
GetPoint	Returns the coordinates of a point along the line.
GetProjectionOfPointOnLine	Computes the projection of a point on a line.

operator=

Copies all the data from another [ELine](#) object into the current [ELine](#) object

[SetFromOriginAndEnd](#)

Sets the geometric parameters (center coordinates, length, and rotation angle) of a [ELine](#) object.

[SetFromTwoPoints](#)

DEPRECATED (you should use [ELine::SetFromOriginAndEnd](#)) : Sets the geometric parameters (center coordinates, length, and rotation angle) of a [ELine](#) object.

ine.CopyTo

Copies all the data of the current [ELine](#) object into another [ELine](#) object and returns it.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ELine CopyTo(
    Euresys.Open_eVision_2_10.ELine other
)
```

Parameters

other

Pointer to the [ELine](#) object in which the current [ELine](#) object data have to be copied.

Remarks

In case of a **NULL** pointer, a new [ELine](#) object will be created and returned.

ELine.ELine

Constructs a [ELine](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ELine(
)

void ELine(
    Euresys.Open_eVision_2_10.EPoint center,
    float length,
    float angle
)

void ELine(
    Euresys.Open_eVision_2_10.EPoint origin,
    Euresys.Open_eVision_2_10.EPoint end
)

void ELine(
    Euresys.Open_eVision_2_10.ELine other
)
```

Parameters

center

Center coordinates of the line at its nominal position. The default value is **(0,0)**.

length

Nominal length of the line. The default value is **100**.

angle

Nominal rotation angle of the line. The default value is **0**.

origin

Origin point coordinates of the line.

end

End point coordinates of the line.

other

Another [ELine](#) object to be copied in the new [ELine](#) object.

ELine.End

End point coordinates of the [ELine](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint End
    { get; }
```

ELine.GetAngleBetweenLines

Computes the angle between two lines.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float GetAngleBetweenLines (
    Euresys.Open_eVision_2_10.ELine line1,
    Euresys.Open_eVision_2_10.ELine line2
)
```

Parameters

line1

First line

line2

Second line

Remarks

The angle returned by this function is signed in the trigonometric sense, meaning that $\text{angle}(1,2) = -\text{angle}(2,1)$.

ELine.GetDistanceBetweenPointAndLine

Computes the distance between a point and a line.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float GetDistanceBetweenPointAndLine(
    Euresys.Open_eVision_2_10.EPoint pt,
    Euresys.Open_eVision_2_10.ELine line,
    bool limited
)
```

Parameters

- pt*
The point.
- line*
The line.
- limited*
Indicates if the line parameter should be considered as an infinite line or as a segment.

ELine.GetIntersectionOfLines

Computes the intersection between two lines.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
int GetIntersectionOfLines(  
    Euresys.Open_eVision_2_10.ELine line1,  
    Euresys.Open_eVision_2_10.ELine line2,  
    Euresys.Open_eVision_2_10.EPoint intersection,  
    bool limited  
)
```

Parameters

line1

First line.

line2

Second line.

intersection

Found intersection.

limited

Indicates if the line parameters should be considered as infinite lines or as a segments.

Remarks

The function returns the number of intersections found. It will return -1 if the two lines are overlapping.

ELine.GetPoint

Returns the coordinates of a point along the line.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint GetPoint(  
    float fraction  
)
```

Parameters

fraction

Point location expressed as a fraction of the line length (range **[-1, +1]**).

ELine.GetProjectionOfPointOnLine

Computes the projection of a point on a line.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint GetProjectionOfPointOnLine(
    Euresys.Open_eVision_2_10.EPoint pt,
    Euresys.Open_eVision_2_10.ELine line
)
```

Parameters

pt

The point.

line

The line.

ELine.Length

Length of the [ELine](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float Length
{ get; set; }
```

Remarks

By default, the length of the line is **100**, which means 100 pixels when the field of view is not calibrated, and 100 physical units in case of a calibrated field of view.

ELine.operator=

Copies all the data from another [ELine](#) object into the current [ELine](#) object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ELine operator=(
    Euresys.Open_eVision_2_10.ELine other
)
```

Parameters

other

[ELine](#) object to be copied

ELine.Org

Origin point coordinates of the [ELine](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint Org
{ get; }
```

ELine.SetFromOriginAndEnd

Sets the geometric parameters (center coordinates, length, and rotation angle) of a [ELine](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetFromOriginAndEnd(
    Euresys.Open_eVision_2_10.EPoint origin,
    Euresys.Open_eVision_2_10.EPoint end
)
```

Parameters

origin

Origin point coordinates of the line.

end

End point coordinates of the line.

ELine.SetFromTwoPoints

DEPRECATED (you should use [ELine::SetFromOriginAndEnd](#)) : Sets the geometric parameters (center coordinates, length, and rotation angle) of a [ELine](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetFromTwoPoints(
    Euresys.Open_eVision_2_10.EPoint origin,
    Euresys.Open_eVision_2_10.EPoint end
)
```

Parameters

origin

Origin point coordinates of the line.

end

End point coordinates of the line.

4.104. ELineGauge Class

Manages a line fitting gauge.

Base Class: [ELineStyle](#)

Namespace: Euresys.Open_eVision_2_10

Properties

[Active](#)

Sets the flag indicating whether the gauge is active or not.

[AverageDistance](#)

Average distance between the sampled points and the fitted model.

[ClippingMode](#)

Clipping mode, that allows to choose how the fitted segment length and center are computed.

[FilteringThreshold](#)

Relative filtering threshold, that is the fraction of the average distance between the sampled points and the fitted model above which a point is filtered out.

[HVConstraint](#)

Status of the restriction on the orientation of the point location gauge or model fitting sample paths.

[KnownAngle](#)

Flag indicating whether the slope of the line to be fitted is known or not.

Line	Sets the nominal position, length and rotation angle of the line fitting gauge, according to a known ELine object.
MeasuredLine	Information pertaining to the fitted line.
MinAmplitude	Offset added to the Threshold when a peak is to be detected.
MinArea	Minimum area value.
NumFilteringPasses	Number of filtering passes for a model fitting operation.
NumMeasuredPoints	Number of edge-crossing points along the sample path of the gauge that was inspected with the last call to ELineGauge::MeasureSample .
NumSamples	Number of sampled points during the model fitting operation.
NumSkipRanges	Number of skip ranges in the gauge after a call to ELineGauge::AddSkipRange .
NumValidSamples	Number of valid sample points remaining after a model fitting operation.
RectangularSamplingArea	Flag indicating whether the sampling area remains a rectangle when rotated, instead of becoming a parallelogram.

SamplingStep	Approximate distance between sampled points during a model fitting operation.
Smoothing	Number of pixels used for the low-pass filtering operation.
Thickness	Number of parallel segments used to extract the data profile.
Threshold	Threshold level used to delimit significant peaks in the data profile.
Tolerance	Searching area half thickness of the line fitting gauge.
TransitionChoice	Transition choice.
TransitionIndex	Index (from 0 on) of the transition to be retained when the transition choice parameter is set to NthFromBegin or NthFromEnd .
TransitionType	Transition type.
Type	Shape type.
Valid	Flag indicating if at least one valid transition has been found.

Methods

AddSkipRange

Adds an item to the set of skip ranges and returns the index of the newly added range.

CopyTo

Copies all the data of the current [ELineGauge](#) object into another [ELineGauge](#) object, and returns it.

Drag

Moves a handle to a new position and updates the position parameters of the gauge.

Draw

Draws a graphical representation of a point location or model fitting gauge, as defined by [EDrawingMode](#).

DrawWithCurrentPen

Draws a graphical representation of a point location or model fitting gauge, as defined by [EDrawingMode](#).

ELineGauge

Constructs a line measurement context.

GetMeasuredPeak

Returns information pertaining to the default derivative peak, along one of the sample paths of the gauge.

GetMeasuredPoint

Returns the coordinates of a sample point, measured along one of the sample paths of the gauge.

GetMinNumFitSamples

Returns the minimum number of samples required for fitting on each side of the shape.

GetSample

Allows to retrieve the sample points found along the line.

GetSkipRange

Allows to retrieve the start and end values of the skip range corresponding to the given index, if it is valid (i.e. if it has previously been created by a call to the [ELineGauge::AddSkipRange](#) method).

HitTest

Checks whether the cursor is positioned over a handle (**TRUE**) or not (**FALSE**).

Measure

Triggers the point location or the model fitting operation.

MeasureSample

Computes the sample points along the sample path whose index in the list is given by the **pathIndex** parameter.

MeasureWithoutFitting

Triggers the point location without line fitting operation.

operator=

Compares the instance with another [ELineGauge](#) object and returns TRUE if they are identical.

Plot

Draws the profile that is crossed by one of the sample paths of the gauge, as defined by [EPlotItem](#).

PlotWithCurrentPen

Draws the profile that is crossed by one of the sample paths of the gauge, as defined by [EPlotItem](#).

Process

Triggers the process pertaining to a shape or gauge and all the daughter gauges attach to it.

RemoveAllSkipRanges

Removes all the skip ranges previously created by a call to [ELineGauge::AddSkipRange](#).

RemoveSkipRange

After a call to [ELineGauge::AddSkipRange](#), removes the skip range with the given index.

SetMinNumFitSamples

┌ Sets the minimum number of samples required for fitting on each
├ side of the shape.
└

ineGauge.Active

Sets the flag indicating whether the gauge is active or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
override bool Active
    { get; set; }
```

Remarks

When complex gauging is required, several gauges can be grouped together. Applying [ELineGauge::Process](#) to the mother gauge or shape triggers the measurement of the whole. Only the active gauges will participate in the process. By default, the gauge is active (**TRUE**).

ELineGauge.AddSkipRange

Adds an item to the set of skip ranges and returns the index of the newly added range.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint AddSkipRange (
    uint start,
    uint end
)
```

Parameters

start

Beginning of the skip range.

end

End of the skip range.

Remarks

The samples indices between start and end (including the boundaries) will be discarded during the measurement process. The [ELineGauge::AddSkipRange](#) method allows to define skip ranges in an [ELineGauge](#). This means that, at measure time, samples belonging to these ranges will not be taken into account. A sample may belong to more than one skip range; to be discarded, a sample has to pertain to at least one range. Moreover, the skip ranges are allowed to overlap one another. The range is allowed to be reversed (i.e. end is not required to be greater than start). Also, start and end are not required to reference valid indices at the time of the call (i.e. the range may lie outside of the current return value for [ELineGauge::NumSamples](#)).

ELineGauge.AverageDistance

Average distance between the sampled points and the fitted model.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float AverageDistance
{ get; }
```

Remarks

Irrelevant in case of a point location operation.

ELineGauge.ClippingMode

Clipping mode, that allows to choose how the fitted segment length and center are computed.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EClippingMode ClippingMode
{ get; set; }
```

Remarks

By default, the clipping mode is [CenteredNominal](#), which corresponds to the behavior appearing in Open eVision version 6.4 and before.

ELineGauge.CopyTo

Copies all the data of the current [ELineGauge](#) object into another [ELineGauge](#) object, and returns it.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
Euresys.Open_eVision_2_10.ELineGauge CopyTo(
    Euresys.Open_eVision_2_10.ELineGauge other,
    bool recursive
)
```

Parameters

other

Pointer to the [ELineGauge](#) object in which the current [ELineGauge](#) object data have to be copied.

recursive

TRUE if the children gauges have to be copied as well, **FALSE** otherwise.

Remarks

In case of a **NULL** pointer, a new [ELineGauge](#) object will be created and returned.

ELineGauge.Drag

Moves a handle to a new position and updates the position parameters of the gauge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Drag(
    int x,
    int y
)
```

Parameters

x

Cursor current X coordinate.

y

Cursor current Y coordinate.

ELineGauge.Draw

Draws a graphical representation of a point location or model fitting gauge, as defined by [EDrawingMode](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingMode

Indicates how the point location or model fitting gauge must be displayed, as defined by [EDrawingMode](#).

daughters

TRUE if the daughters gauges are to be displayed also.

color

The color in which to draw the overlay.

ELineGauge.DrawWithCurrentPen

Draws a graphical representation of a point location or model fitting gauge, as defined by [EDrawingMode](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawWithCurrentPen(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingMode

Indicates how the point location or model fitting gauge must be displayed, as defined by [EDrawingMode](#).

daughters

TRUE if the daughters gauges are to be displayed also.

ELineGauge.ELineGauge

Constructs a line measurement context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ELineGauge(
)
```

```
void ELineGauge (  
    Euresys.Open_eVision_2_10.ELineGauge other  
)
```

Parameters

other

Another [ELineGauge](#) object to be copied in the new [ELineGauge](#) object.

Remarks

With the default constructor, all the parameters are initialized to their respective default values. With the copy constructor, the constructed line measurement context is based on a pre-existing [ELineGauge](#) object. The gauge children are also copied. Hierarchy copying through a copy constructor is always recursive. To disable this recursion, use instead the [ELineGauge::CopyTo](#) method.

ELineGauge.FilteringThreshold

Relative filtering threshold, that is the fraction of the average distance between the sampled points and the fitted model above which a point is filtered out.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
float FilteringThreshold  
  
    { get; set; }
```

Remarks

Irrelevant in case of a point location operation. During a model fitting operation, the "filtering" process can be invoked to remove outliers, i.e. points that were located significantly far away from the fitted model (so that their position is dubious).

ELineGauge.GetMeasuredPeak

Returns information pertaining to the default derivative peak, along one of the sample paths of the gauge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPeak GetMeasuredPeak(  
    uint index  
)
```

Parameters

index

This argument must be left unchanged from its default value, i.e. **~0** (= **0xFFFFFFFF**).

Remarks

[ELineGauge::GetMeasuredPeak](#) returns the information about the derivative peak that meets the following two requirements: 1. It lies on the sample path inspected with the last call to [ELineGauge::MeasureSample](#), and 1. It is associated with the edge-crossing point along the latter sample path that is selected by the transition choice parameter (cf. [ELineGauge::TransitionChoice](#)).

Note. For this method to succeed, it is necessary to previously call [ELineGauge::MeasureSample](#).

ELineGauge.GetMeasuredPoint

Returns the coordinates of a sample point, measured along one of the sample paths of the gauge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EPoint GetMeasuredPoint(  
    uint index  
)
```

Parameters

index

This argument must be left unchanged from its default value, i.e. **~0** (= **0xFFFFFFFF**).

Remarks

These coordinates pertain to the world space; they are expressed in the reference frame to which the current `ELineGauge` object belongs. The gauging process uses a list of sample points to find the shape position and size that best fit a given image. These sample points are measured along the sample paths defined by the gauge geometry. [ELineGauge::GetMeasuredPoint](#) returns the coordinates of the sample point that meets the following two requirements:

1. It lies on the sample path inspected with the last call to [ELineGauge::MeasureSample](#), and
1. Among all the sample points along the latter sample path, it is the one selected by the [ELineGauge::TransitionChoice](#) property.

Note. For this method to succeed, it is necessary to previously call [ELineGauge::MeasureSample](#).

ELineGauge.GetMinNumFitSamples

Returns the minimum number of samples required for fitting on each side of the shape.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void GetMinNumFitSamples (  
    out int side0,  
    out int side1,  
    out int side2,  
    out int side3  
)
```

Parameters

side0

Minimum number of samples on the top side of the rectangle.

side1

Minimum number of samples on the left side of the rectangle.

side2

Minimum number of samples on the bottom side of the rectangle.

side3

Minimum number of samples on the right side of the rectangle.

Remarks

Irrelevant in case of a point location operation.

ELineGauge.GetSample

Allows to retrieve the sample points found along the line.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool GetSample(
    Euresys.Open_eVision_2_10.EPoint pt,
    uint index
)

void GetSample(
    Euresys.Open_eVision_2_10.ESamplePoint pt,
    uint index
)
```

Parameters

pt

[EPoint](#) structure that will contain the sample position.

index

The sample index

Remarks

The method provides the sample point corresponding to the supplied index. The returned value corresponds to the sample validity.

ELineGauge.GetSkipRange

Allows to retrieve the start and end values of the skip range corresponding to the given index, if it is valid (i.e. if it has previously been created by a call to the [ELineGauge::AddSkipRange](#) method).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetSkipRange(
    uint index,
    out uint start,
    out uint end
)
```

Parameters

index

Index of the skip range.

start

Beginning of the skip range.

end

End of the skip range.

Remarks

Start is guaranteed to be smaller or equal to end.

ELineGauge.HitTest

Checks whether the cursor is positioned over a handle (**TRUE**) or not (**FALSE**).

Namespace: Euresys.Open_eVision_2_10


```
[C#]
bool HitTest(
    bool daughters
)
```

Parameters

daughters

TRUE if the daughters gauges handles have to be considered as well.

ELineGauge.HVConstraint

Status of the restriction on the orientation of the point location gauge or model fitting sample paths.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool HVConstraint
{ get; set; }
```

Remarks

Sample paths are the point location gauges placed along the model to be fitted.

ELineGauge.KnownAngle

Flag indicating whether the slope of the line to be fitted is known or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
bool KnownAngle
```

```
{ get; set; }
```

Remarks

A line model to be fitted may have a well-known slope. It is possible to impose the value of this slope, thus removing one degree of freedom. The line fitting gauge slope is set by means of [ELineGauge](#). The sign of the rotation angle depends whether the field of view is calibrated or not. * When the field of view is calibrated, the coordinate system is said to be direct, the abscissa extends rightwards and the ordinate extends upwards. In this case, an anticlockwise rotation leads to a positive angle value. * When the field of view is not calibrated, the coordinate system is said to be inverse, the abscissa extends rightwards and the ordinate extends downwards. In this case, a clockwise rotation leads to a positive angle value.

ELineGauge.Line

Sets the nominal position, length and rotation angle of the line fitting gauge, according to a known [ELine](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
override Euresys.Open_eVision_2_10.ELine Line
```

```
{ get; set; }
```

ELineGauge.Measure

Triggers the point location or the model fitting operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Measure (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage
)
```

Parameters

sourceImage

Pointer to the source image.

Remarks

When this method is called, and if not enough valid points are detected, then the method returns directly, and the measured gauge is set to the nominal parameters.

ELineGauge.MeasuredLine

Information pertaining to the fitted line.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ELine MeasuredLine
{ get; }
```

ELineGauge.MeasureSample

Computes the sample points along the sample path whose index in the list is given by the **pathIndex** parameter.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void MeasureSample (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    uint pathIndex
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

pathIndex

Sample path index.

Remarks

This method stores its results into a temporary variable inside the ELineGauge object.

ELineGauge.MeasureWithoutFitting

Triggers the point location without line fitting operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void MeasureWithoutFitting (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage
)
```

Parameters

sourceImage

Source image.

Remarks

This method performs the actual measurement for each transition, but does not perform the line fitting. This means that individual samples will be available through the [ELineGauge::GetSample](#) method, but the gauge position will not be changed. Please note that the filtering will not be performed in this method, since it relies upon the fitting process. The filtering parameters will thus be unused.

ELineGauge.MinAmplitude

Offset added to the **Threshold** when a peak is to be detected.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint MinAmplitude  
  
{ get; set; }
```

Remarks

When analyzing a derivative profile, a peak is made up of consecutive pixel values above **Threshold**. To detect weak [strong] transitions, lower [raise] the **Threshold** value. To avoid interference of noise, an additional parameter is provided. The **MinAmplitude** parameter is an offset added to **Threshold** when a peak is to be detected. When the pixel values of the derivative profile do not reach **Threshold + MinAmplitude**, the peak is not taken into account. Anyway, when a peak is taken into account, all the pixels with values above **Threshold** are considered (for more accuracy). Setting the **MinAmplitude** value to **0** merely cancels its effect.

ELineGauge.MinArea

Minimum area value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint MinArea  
  
{ get; set; }
```

Remarks

A transition is detected if its derivative peak reaches **Threshold + MinAmplitude** value, and then declared valid if the area between the peak curve and the horizontal at level **Threshold** reaches the **MinArea** value.

ELineGauge.NumFilteringPasses

Number of filtering passes for a model fitting operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint NumFilteringPasses  
  
    { get; set; }
```

Remarks

Irrelevant in case of a point location operation. During a filtering pass, the points that are too distant from the model are discarded. During a model fitting operation, the "filtering" process can be invoked to remove outliers, i.e. points that were located significantly far away from the fitted model (so that their position is dubious). By default (the number of filtering passes is 0), the outliers rejection process is disabled.

ELineGauge.NumMeasuredPoints

Number of edge-crossing points along the sample path of the gauge that was inspected with the last call to [ELineGauge::MeasureSample](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint NumMeasuredPoints  
  
    { get; }
```

Remarks

Note. For this method to succeed, it is necessary to previously call [ELineGauge::MeasureSample](#).

ELineGauge.NumSamples

Number of sampled points during the model fitting operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint NumSamples  
  
    { get; }
```

Remarks

Irrelevant in case of a point location operation. After a model fitting operation, a number of points have been fitted along the model. Among them, some are not reliable because of their **Area** value. Among the remaining ones, some are filtered out (**NumFilteringPasses**, **FilteringThreshold**).

ELineGauge.NumSkipRanges

Number of skip ranges in the gauge after a call to [ELineGauge::AddSkipRange](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint NumSkipRanges  
  
    { get; }
```

ELineGauge.NumValidSamples

Number of valid sample points remaining after a model fitting operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
uint NumValidSamples  
  
    { get; }
```

Remarks

Irrelevant in case of a point location operation. After a model fitting operation, a number of points have been fitted along the model. Among them, some are not reliable because of their **Area** value. Among the remaining ones, some are filtered out (**NumFilteringPasses**, **FilteringThreshold**).

ELineGauge.operator=

Compares the instance with another [ELineGauge](#) object and returns TRUE if they are identical.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
Euresys.Open_eVision_2_10.ELineGauge operator=(  
    Euresys.Open_eVision_2_10.ELineGauge other  
)
```

Parameters

other
[ELineGauge](#) object to be compared

ELineGauge.Plot

Draws the profile that is crossed by one of the sample paths of the gauge, as defined by [EPlotItem](#).

Namespace: Euresys.Open_eVision_2_10


```

[C#]
void Plot(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EPlotItem drawItems,
    float width,
    float height,
    float originX,
    float originY
)

void Plot(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.EPlotItem drawItems,
    float width,
    float height,
    float originX,
    float originY
)

void Plot(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.EPlotItem drawItems,
    float width,
    float height,
    float originX,
    float originY
)

```

Parameters

graphicContext

Handle of the device context on which to draw.

drawItems

Boolean combination of [EPlotItem](#) members, that indicates which items must be displayed.

width

Width of the plot.

height

Height of the plot.

originX

Origin point coordinates of the plot along the X axis.

originY

Origin point coordinates of the plot along the Y axis.

color

The color in which to draw the overlay.

Remarks

The sample path that is taken into considered is the one inspected with the last call to [ELineGauge::MeasureSample](#).

Note. For this method to succeed, it is necessary to previously call [ELineGauge::MeasureSample](#).

ELineGauge.PlotWithCurrentPen

Draws the profile that is crossed by one of the sample paths of the gauge, as defined by [EPlotItem](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void PlotWithCurrentPen(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EPlotItem drawItems,
    float width,
    float height,
    float originX,
    float originY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawItems

Boolean combination of [EPlotItem](#) members, that indicates which items must be displayed.

width

Width of the plot.

height

Height of the plot.

originX

Origin point coordinates of the plot along the X axis.

originY

Origin point coordinates of the plot along the Y axis.

Remarks

The sample path that is taken into considered is the one inspected with the last call to [ELineGauge::MeasureSample](#).

Note. For this method to succeed, it is necessary to previously call [ELineGauge::MeasureSample](#).

ELineGauge.Process

Triggers the process pertaining to a shape or gauge and all the daughter gauges attach to it.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Process (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    bool daughters
)
```

Parameters

sourceImage

Pointer to the BW8 roi source image.

daughters

Flag indicating whether the daughters shapes inherit of the same behavior.

Remarks

When complex gauging is required, several gauges can be grouped together. Applying **Process** to the mother gauge or shape triggers the measurement of the whole. Only the active gauges will participate in the process.

ELineGauge.RectangularSamplingArea

Flag indicating whether the sampling area remains a rectangle when rotated, instead of becoming a parallelogram.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
bool RectangularSamplingArea  
  
{ get; set; }
```

Remarks

By default, this flag is set to **TRUE**: the sampling area always remains a rectangle. Setting this property is only useful when the thickness transition parameter is greater than 1. In fact, when thickness transition parameter is equal to 1, rectangle and parallelogram reduce to the same segment.

ELineGauge.RemoveAllSkipRanges

Removes all the skip ranges previously created by a call to [ELineGauge::AddSkipRange](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void RemoveAllSkipRanges (  
)
```

ELineGauge.RemoveSkipRange

After a call to [ELineGauge::AddSkipRange](#), removes the skip range with the given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void RemoveSkipRange (  
    uint index  
)
```

Parameters

index

Index of the skip range to remove, as returned by [ELineGauge::AddSkipRange](#).

ELineGauge.SamplingStep

Approximate distance between sampled points during a model fitting operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float SamplingStep  
{ get; set; }
```

Remarks

Irrelevant in case of a point location operation. To fit a model, a series of point location operations are performed along the model. The point location gauges spacing is given by the sampling step. By default, the sampling step is set to **5**, which means 5 pixels when the field of view is not calibrated, and 5 physical units in case of a calibrated field of view. Be aware that if the sampling step is too large, the number of sampled points along the model will not be sufficient enough to accurately locate it.

ELineGauge.SetMinNumFitSamples

Sets the minimum number of samples required for fitting on each side of the shape.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetMinNumFitSamples (
    int side0,
    int side1,
    int side2,
    int side3
)
```

Parameters

side0

Required number of samples to correctly fit the line. The default value is **2**. It is the only parameter taken into account.

side1

Not used.

side2

Not used.

side3

Not used.

Remarks

Irrelevant in case of a point location operation. When the [ELineGauge::Measure](#) method is called, and if not enough valid points are detected, then the method returns directly, and the measured gauge is set to the nominal parameters.

ELineGauge.Smoothing

Number of pixels used for the low-pass filtering operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint Smoothing
    { get; set; }
```

Remarks

To reduce the effect of noise, the profile data can be low-pass filtered along the point location gauge direction.

ELineGauge.Thickness

Number of parallel segments used to extract the data profile.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint Thickness  
    { get; set; }
```

Remarks

To reduce the effect of noise and/or strengthen a transition, several parallel profiles can be accumulated.

ELineGauge.Threshold

Threshold level used to delimit significant peaks in the data profile.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint Threshold  
    { get; set; }
```

Remarks

When analyzing a derivative profile, a peak is made up of consecutive pixel values above **Threshold**. To detect weak [strong] transitions, lower [raise] the **Threshold** value. To avoid interference of noise, an additional parameter is provided. The **MinAmplitude** parameter is an offset added to **Threshold** when a peak is to be detected. When the pixel values of the derivative profile do not reach **Threshold + MinAmplitude**, the peak is not taken into account. Anyway, when a peak is taken into account, all the pixels with values above **Threshold** are considered (for more accuracy). Setting the **MinAmplitude** value to **0** merely cancels its effect.

ELineGauge.Tolerance

Searching area half thickness of the line fitting gauge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
float Tolerance  
  
{ get; set; }
```

Remarks

By default, the searching area thickness of the line fitting gauge is **20** (2x10), which means 20 pixels when the field of view is not calibrated, and 20 physical units in case of a calibrated field of view.

ELineGauge.TransitionChoice

Transition choice.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
Euresys.Open_eVision_2_10.ETransitionChoice TransitionChoice
```



```
{ get; set; }
```

Remarks

Several peaks may be detected along a point location gauge. This parameter helps to select the desired transition. In case of [NthFromBegin](#) or [NthFromEnd](#) transition choice, set [ELineGauge::TransitionIndex](#) to specify the desired transition. By default, the selected transition corresponds to the one with the largest amplitude ([LargestAmplitude](#)).

ELineGauge.TransitionIndex

Index (from **0** on) of the transition to be retained when the transition choice parameter is set to [NthFromBegin](#) or [NthFromEnd](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint TransitionIndex  
  
{ get; set; }
```

Remarks

Several peaks may be detected along a point location gauge. This parameter helps to select the desired transition. By default, the first transition is retained (the index value is **0**).

ELineGauge.TransitionType

Transition type.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.ETransitionType TransitionType  
  
{ get; set; }
```

Remarks

The type of a transition tells whether it crosses increasing or decreasing gray-level values. This helps discriminate between nearby edges of an object. By default, the searched transition type is indifferently a black to white or a white to black transition ([BwOrWb](#)).

ELineGauge.Type

Shape type.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
override Euresys.Open_eVision_2_10.EShapeType Type  
  
{ get; }
```

ELineGauge.Valid

Flag indicating if at least one valid transition has been found.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
bool Valid
```

```
{ get; }
```

Remarks

A **FALSE** value means that no measurement has been performed. A **TRUE** value means that a transition was found along the sample path inspected with the last call to [ELineGauge::MeasureSample](#), and thus a point has been measured.

4.105. ELineShape Class

Manages a line shape.

Base Class: [EShape](#)

Derived Class(es): [ELineGauge](#)

Namespace: Euresys.Open_eVision_2_10

Properties

[Angle](#)

Orientation of the shape.

[Center](#)

Center point of the frame.

[CenterX](#)

Abscissa of the origin point of the frame.

[CenterY](#)

Ordinate of the origin point of the frame.

[End](#)

End point coordinates of the [ELineShape](#) object.

Length

Length of the [ELineShape](#) object.

Line

Sets the nominal position, length and rotation angle of the line, according to a known [ELine](#) object.

Org

Origin point coordinates of the [ELineShape](#) object.

Scale

Horizontal sensor resolution, in pixels per unit.

Type

M Shape type.

e

Methods

Closest

Find the daughter shape that is the closest to this shape. To retrieve the closest shape, use [EShape::ClosestShape](#).

CopyTo

Copies all the data of the current [ELineShape](#) object into another [ELineShape](#) object and returns it.

Drag

Moves a handle to a new position and updates the position parameters of the shape.

Draw

Draws a graphical representation of a shape, as defined by [EDrawingMode](#).

[DrawWithCurrentPen](#)

Draws a graphical representation of a shape, as defined by [EDrawingMode](#).

[GetPoint](#)

Returns the coordinates of a point along the line.

[HitTest](#)

Checks if there is a handle under the cursor.

[operator=](#)

Assignment operator

[SetCenterXY](#)

Sets the center coordinates of a [ELineShape](#) object.

[SetFromOriginAndEnd](#)

Sets the geometric parameters (center coordinates, length, and rotation angle) of a [ELineShape](#) object.

[SetFromTwoPoints](#)

DEPRECATED (you should use [ELineShape::SetFromOriginAndEnd](#))
: Sets the geometric parameters (center coordinates, length, and rotation angle) of a [ELineShape](#) object.

ineShape.Angle

Orientation of the shape.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
float Angle
```

```
{ get; set; }
```

ELineStyle.Center

Center point of the frame.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EPoint Center
```

```
{ get; set; }
```

ELineStyle.CenterX

Abscissa of the origin point of the frame.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float CenterX
```

```
{ get; }
```

ELineStyle.CenterY

Ordinate of the origin point of the frame.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float CenterY  
    { get; }
```

ELineShape.Closest

Find the daughter shape that is the closest to this shape. To retrieve the closest shape, use [EShape::ClosestShape](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Closest(  
    )
```

ELineShape.CopyTo

Copies all the data of the current [ELineShape](#) object into another [ELineShape](#) object and returns it.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.ELineShape CopyTo(  
    Euresys.Open_eVision_2_10.ELineShape dest,  
    bool bRecursive  
)
```

Parameters

dest

Pointer to the [ELineShape](#) object in which the current [ELineShape](#) object data have to be copied.

bRecursive

TRUE if the children shapes have to be copied as well, **FALSE** otherwise.

Remarks

In case of a **NULL** pointer, a new [ELineShape](#) object will be created and returned.

ELineShape.Drag

Moves a handle to a new position and updates the position parameters of the shape.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void Drag(  
    int n32CursorX,  
    int n32CursorY  
)
```

Parameters

n32CursorX

Current cursor coordinates.

n32CursorY

Current cursor coordinates.

ELineStyle.Draw

Draws a graphical representation of a shape, as defined by [EDrawingMode](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingMode

Indicates how the point location or model fitting gauge must be displayed, as defined by [EDrawingMode](#).

daughters

TRUE if the daughters gauges are to be displayed also.

color

-

ELineStyle.DrawWithCurrentPen

Draws a graphical representation of a shape, as defined by [EDrawingMode](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawWithCurrentPen(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingMode

Indicates how the point location or model fitting gauge must be displayed, as defined by [EDrawingMode](#).

daughters

TRUE if the daughters gauges are to be displayed also.

ELineStyle.End

End point coordinates of the [ELineStyle](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint End
{ get; }
```

ELineStyle.GetPoint

Returns the coordinates of a point along the line.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint GetPoint(  
    float fraction  
)
```

Parameters

fraction

Point location expressed as a fraction of the line length (range **[-1, +1]**).

ELineStyle.HitTest

Checks if there is a handle under the cursor.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool HitTest(  
    bool bDaughters  
)
```

Parameters

bDaughters

Indicates if the check must be done in the whole hierarchy or just this object.

ELineStyle.Length

Length of the [ELineStyle](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Length  
  
{ get; set; }
```

Remarks

By default, the length of the line is **100**, which means 100 pixels when the field of view is not calibrated, and 100 physical units in case of a calibrated field of view.

ELineStyle.Line

Sets the nominal position, length and rotation angle of the line, according to a known [ELine](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
virtual Euresys.Open_eVision_2_10.ELine Line  
  
{ get; set; }
```

ELineStyle.operator=

Assignment operator

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.ELineShape operator=(  
    Euresys.Open_eVision_2_10.ELineShape other  
)
```

Parameters

other

Reference to the [ELineShape](#) object used for the assignment

ELineShape.Org

Origin point coordinates of the [ELineShape](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint Org  
{ get; }
```

ELineShape.Scale

Horizontal sensor resolution, in pixels per unit.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float Scale
```

```
{ get; set; }
```

ELineStyle.SetCenterXY

Sets the center coordinates of a [ELineStyle](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void SetCenterXY(  
    float centerX,  
    float centerY  
)
```

Parameters

centerX

Center coordinates of the [ELineStyle](#) object.

centerY

Center coordinates of the [ELineStyle](#) object.

ELineStyle.SetFromOriginAndEnd

Sets the geometric parameters (center coordinates, length, and rotation angle) of a [ELineStyle](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetFromOriginAndEnd(
    Euresys.Open_eVision_2_10.EPoint origin,
    Euresys.Open_eVision_2_10.EPoint end
)
```

Parameters

origin

Origin point coordinates of the line.

end

End point coordinates of the line.

ELineStyle.SetFromTwoPoints

DEPRECATED (you should use [ELineStyle::SetFromOriginAndEnd](#)) : Sets the geometric parameters (center coordinates, length, and rotation angle) of a [ELineStyle](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetFromTwoPoints(
    Euresys.Open_eVision_2_10.EPoint origin,
    Euresys.Open_eVision_2_10.EPoint end
)
```

Parameters

origin

Origin point coordinates of the line.

end

End point coordinates of the line.

ELineStyle.Type

Shape type.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
override Euresys.Open_eVision_2_10.EShapeType Type
{ get; }
```

4.106. EListItem Class

Describes list items. This class pertains to the EasyObject legacy API. Please use [ECodedImage2](#) for all new developments instead.

Remarks

A list is a sequence of orderly list items. Each list item contains a pointer to a memory zone containing its data, a pointer to the previous list item, and a pointer to the next list item. List itemsA few [ECodedImage](#) methods handle EListItem objects, or EListItem pointers. Runs lists[ECodedImage::GetFirstRunData](#), [ECodedImage::GetFirstRunPtr](#), [ECodedImage::GetLastRunData](#), [ECodedImage::GetLastRunPtr](#), [ECodedImage::GetPreviousRunData](#), [ECodedImage::GetPreviousRunPtr](#), [ECodedImage::GetNextRunData](#), [ECodedImage::GetNextRunPtr](#) These properties and methods allow to traverse the runs lists from the first run to the last, or from one run to its previous or next neighbor. A run can also be directly reached by its index within the list. The first run has index **0**. The last run has index **NumRuns-1**. The [ECodedImage::GetRunData](#) and [ECodedImage::GetRunDataPtr](#) methods return the run data, or a pointer to the run data. Objects lists[ECodedImage::GetFirstObjData](#), [ECodedImage::GetLastObjData](#), [ECodedImage::GetPreviousObjData](#), [ECodedImage::GetPreviousObjPtr](#), [ECodedImage::GetNextObjData](#), [ECodedImage::GetNextObjPtr](#) These properties and methods allow to traverse the objects lists from the first object to the last, or from one object to its previous or next neighbor. An object can also be directly reached by its index within the list. The first object has index **0**. The last object has index **NumObjects-1**. The [ECodedImage::GetObjectData](#) and [ECodedImage::GetObjectDataPtr](#) methods return the object data, or a pointer to the object data.

Namespace: Euresys.Open_eVision_2_10

4.107. EMailBarcode Class

Manages a complete context for a Mail Barcode.

Namespace: Euresys.Open_eVision_2_10

Properties

ChecksumOk

Returns if the checksum was successfully validated.

ComponentStrings

Returns the list of semantic parts included in the mail barcode text.

Orientation

Returns the orientation of the mail barcode.

Position

Returns the position of the mail barcode in the Image/ROI.

Symbology

Returns the symbology of the mail barcode.

Text

M Returns the full decoded text of the mail barcode.

e

Methods

Draw

Draws the bounding box of the mail barcode.

EmailBarcode

Constructs an EmailBarcodeReader context.

operator=

Assignment operator

M

ailBarcode.ChecksumOk

Returns if the checksum was successfully validated.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool ChecksumOk  
{ get; }
```

EmailBarcode.ComponentStrings

Returns the list of semantic parts included in the mail barcode text.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EStringPair[] ComponentStrings  
{ get; }
```

EmailBarcode.Draw

Draws the bounding box of the mail barcode.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Draw(
    IntPtr hDC,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter adapter,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

Parameters

hDC

Handle of the device context on which to draw.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

adapter

-

EmailBarcode.EmailBarcode

Constructs an EmailBarcodeReader context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EmailBarcode(
)
void EmailBarcode(
    Euresys.Open_eVision_2_10.EmailBarcode other
)
```

Parameters

other

-

EmailBarcode.operator=

Assignment operator

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EmailBarcode operator=(
    Euresys.Open_eVision_2_10.EmailBarcode other
)
```

Parameters

other

-

EmailBarcode.Orientation

Returns the orientation of the mail barcode.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EmailBarcodeOrientation Orientation  
{ get; }
```

EmailBarcode.Position

Returns the position of the mail barcode in the Image/ROI.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.ERectangle Position  
{ get; }
```

EmailBarcode.Symbology

Returns the symbology of the mail barcode.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EmailBarcodeSymbologies Symbology
{ get; }
```

EmailBarcode.Text

Returns the full decoded text of the mail barcode.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
string Text
{ get; }
```

4.108. EmailBarcodeReader Class

Manages a complete context for a Mail Barcode Reader.

Namespace: Euresys.Open_eVision_2_10

Properties

[EnableClutteredBarcodes](#)

Enables cluttered barcode (barcode with fused bars) support.

[EnableDottedBarcodes](#)

Enables dotted barcode support.

ExpectedOrientations

Expected barcode orientations for the Mail Barcode detection.

ExpectedSymbologies

Expected symbologies for the Mail Barcode detection.

ValidateChecksum

M Configures the reader to return barcodes even if their checksum is incorrect.

e

thods

EmailBarcodeReader

Constructs an EmailBarcodeReader context.

Load

Loads the settings of the [EmailBarcodeReader](#) object, from disk.

operator=

Assignment operator

Read

Locates and decodes mail barcodes.

Save

E Saves the current settings of the [EmailBarcodeReader](#) object.

M

ailBarcodeReader.EmailBarcodeReader

Constructs an EmailBarcodeReader context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EMailBarcodeReader (
)
void EMailBarcodeReader (
    Euresys.Open_eVision_2_10.EMailBarcodeReader other
)
```

Parameters

other

-

EMailBarcodeReader.EnableClutteredBarcodes

Enables cluttered barcode (barcode with fused bars) support.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool EnableClutteredBarcodes
{ get; set; }
```

EMailBarcodeReader.EnableDottedBarcodes

Enables dotted barcode support.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```



```
bool EnableDottedBarcodes
```

```
{ get; set; }
```

EmailBarcodeReader.ExpectedOrientations

Expected barcode orientations for the Mail Barcode detection.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
int ExpectedOrientations
```

```
{ get; set; }
```

Remarks

The value is a combination of the members of the [EmailBarcodeOrientation](#) enumerate.

EmailBarcodeReader.ExpectedSymbologies

Expected symbologies for the Mail Barcode detection.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
int ExpectedSymbologies
```

```
{ get; set; }
```

Remarks

The value is a combination of the members of the [EmailBarcodeSymbologies](#) enumerate.

EmailBarcodeReader.Load

Loads the settings of the [EmailBarcodeReader](#) object, from disk.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Load(
    Euresys.Open_eVision_2_10.ESerializer path
)
```

Parameters

path

A string containing the full path to the file.

EmailBarcodeReader.operator=

Assignment operator

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EmailBarcodeReader operator=(
    Euresys.Open_eVision_2_10.EmailBarcodeReader other
)
```

Parameters

other

-

EmailBarcodeReader.Read

Locates and decodes mail barcodes.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EmailBarcode[] Read(
    Euresys.Open_eVision_2_10.EROIBW8 roi
)
```

Parameters

roi

The ROI/Image in which to search for mail barcodes.

Remarks

This method returns the list of the detected barcodes.

EmailBarcodeReader.Save

Saves the current settings of the [EmailBarcodeReader](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer path
)
```

Parameters

path

A string containing the full path to the file.

Remarks

It is advised to use a file extension that is non-standard (for instance *.mbr).

EmailBarcodeReader.ValidateChecksum

Configures the reader to return barcodes even if their checksum is incorrect.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool ValidateChecksum
{ get; set; }
```

4.109. EMatcher Class

Manages a complete matching context in EasyMatch.

Remarks

A matching context consists of a learned pattern and of the parameters required to locate one or more instances of the pattern in a search field.

Namespace: Euresys.Open_eVision_2_10

Properties

[AdvancedLearning](#)

Toggle advanced learning.

[AngleStep](#)

Current angle step.

ContrastMode

Contrast mode.

CorrelationMode

Correlation mode.

DontCareThreshold

"Don't care" threshold.

FilteringMode

Filtering mode.

FinalReduction

Index of the last reduction.

InitialMinScore

Minimum score applied as a selection criterion in the early stages of the matching process.

Interpolate

Interpolation mode.

IsotropicScale

Flag indicating whether isotropic (as opposed to anisotropic) scaling is used.

MaxAngle

Maximum angle, in the current angle unit.

MaxInitialPositions

Maximum number of positions at the first stage of the matching process.

MaxPositions

Maximum number of positions.

MaxScale	Maximum scale factor for isotropic scaling.
MaxScaleX	Maximum horizontal scale factor for anisotropic scaling.
MaxScaleY	Maximum vertical scale factor for anisotropic scaling.
MinAngle	Minimum angle, in the current angle unit.
MinReducedArea	Minimum reduced area parameter.
MinScale	Minimum scale factor for isotropic scaling.
MinScaleX	Minimum horizontal scale factor for anisotropic scaling.
MinScaleY	Minimum vertical scale factor for anisotropic scaling.
MinScore	Minimum score.
NumPositions	Number of good matches found, as defined by EMatcher::MinScore and EMatcher::MaxPositions properties.
NumReductions	Number of reduction steps used in the matching process.
PatternHeight	Learnt pattern height.

PatternLearnt	Returns TRUE after a learning operation has been successfully performed, indicating that the EMatcher object is ready for matching, and FALSE otherwise.
PatternType	Pixel type of the learnt pattern.
PatternWidth	Learnt pattern width.
Positions	Returns a vector of EMatchPosition objects, each containing the position coordinates and other matching results.
ScaleStep	Current value of scale step.
ScaleXStep	Current value of scale X step.
ScaleYStep	Current value of scale Y step.
Version	M Version number of the EMatcher object.

e

Methods

ClearImage	Releases the pointer to the image object that has been passed to the EMatcher object.
------------	---

[CopyLearntPattern](#)

Copies the learnt pattern in the supplied image. If no pattern has been learned, an exception with code [NoPatternLearnt](#) will be thrown.

[CopyTo](#)

Copies all the data of the current [EMatcher](#) object into another [EMatcher](#) object and returns it.

[DrawPosition](#)

Draws a graphical representation of a given occurrence of the pattern in the image, using a rectangle and possibly a small line segment in the upper-left corner.

[DrawPositions](#)

Draws a graphical representation of all occurrences of the pattern in the image, using a rectangle and possibly a small line segment in the upper-left corner.

[DrawPositionsWithCurrentPen](#)

Draws a graphical representation of all occurrences of the pattern in the image, using a rectangle and possibly a small line segment in the upper-left corner.

[DrawPositionWithCurrentPen](#)

Draws a graphical representation of a given occurrence of the pattern in the image, using a rectangle and possibly a small line segment in the upper-left corner.

[EMatcher](#)

Constructs a matching context.

[GetPixelDimensions](#)

Gets the physical pixel dimensions.

GetPosition

Returns an [EMatchPosition](#) object containing the position coordinates.

LearnPattern

Learns a pattern to subsequently match in an image.

Load

Loads the [EMatcher](#). The given [ESerializer](#) must have been created for reading.

Match

Matches the pattern against an image.

operator=

Copies all the data from another [EMatcher](#) object into the current [EMatcher](#) object

Save

Saves the [EMatcher](#). The given [ESerializer](#) must have been created for writing.

SetExtension

Sets the extension of the matching ROI, i.e. puts the horizontal and vertical distances, in pixels, that the found pattern occurrences may fall outside the matching ROI. Such occurrences partially outside the ROI have their score corrected by the ratio between the pattern area outside the ROI and the pattern area inside.

SetPixelDimensions

Sets the physical pixel dimensions.

EMatcher.AdvancedLearning

Toggle advanced learning.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool AdvancedLearning
    { get; set; }
```

Remarks

When enable, the advanced learning process will try to optimize learning parameters like the Minimum Reduced Area. The learning will take more time (from 1x to 5x longer) but the matching probability could be improved. The improvement strongly depends on the pattern source image. The advanced learning is automatically disabled when the method [EMatcher::MinReducedArea](#) is called.

EMatcher.AngleStep

Current angle step.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float AngleStep
    { get; }
```

EMatcher.ClearImage

Releases the pointer to the image object that has been passed to the [EMatcher](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void ClearImage(  
)
```

Remarks

It is the way to tell to the [EMatcher](#) object that its pointer is not valid anymore. The [EMatcher::Match](#) method keeps a copy of the image pointer given as parameter. So, if the user deletes this pointer, the [EMatcher](#) object should be informed.

EMatcher.ContrastMode

Contrast mode.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EMatchContrastMode ContrastMode  
  
{ get; set; }
```

Remarks

By default, the contrast mode is set to [Normal](#).

EMatcher.CopyLearntPattern

Copies the learnt pattern in the supplied image. If no pattern has been learned, an exception with code [NoPatternLearnt](#) will be thrown.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void CopyLearntPattern(
    Euresys.Open_eVision_2_10.EImageBW8 image
)
void CopyLearntPattern(
    Euresys.Open_eVision_2_10.EImageC24 image
)
```

Parameters

image

Pointer to the image in which the learnt pattern will be returned.

EMatcher.CopyTo

Copies all the data of the current [EMatcher](#) object into another [EMatcher](#) object and returns it.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EMatcher CopyTo(
    Euresys.Open_eVision_2_10.EMatcher other
)
```

Parameters

other

Pointer to the [EMatcher](#) object in which the current [EMatcher](#) object parameters are to be copied. If **NULL** (default), a new [EMatcher](#) object will be created and returned.

EMatcher.CorrelationMode

Correlation mode.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.ECorrelationMode CorrelationMode  
{ get; set; }
```

Remarks

This property tells what normalization rule is used to correlate the pattern to the image. By default, the correlation mode is set to [Normalized](#).

EMatcher.DontCareThreshold

"Don't care" threshold.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint DontCareThreshold  
{ get; set; }
```

Remarks

If the pattern cannot be inscribed in a rectangle because there are foreign objects in a close neighborhood, mismatches can be avoided by using "don't care" pixels: all the pattern pixels whose value is strictly below **DontCareThreshold** will be ignored. By default, this property is set to **0**: no "don't care" pixel exists.

Note. When you use the "don't care" feature, either the pattern is well contrasted from its background -then set **DontCareThreshold** to an appropriate thresholding value- or it is not -then set the background pixels of the pattern to some low value (by a masking operation) and set **DontCareThreshold** to this low value plus one.

EMatcher.DrawPosition

Draws a graphical representation of a given occurrence of the pattern in the image, using a rectangle and possibly a small line segment in the upper-left corner.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawPosition(
    IntPtr graphicContext,
    uint index,
    bool bCorner,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void DrawPosition(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    uint index,
    bool bCorner,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

```
void DrawPosition(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    uint index,  
    bool bCorner,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

index

Occurrence index, in range **0..NumPositions-1**.

bCorner

TRUE if the corner mark is to be drawn. **FALSE** by default. (This mark is useful when large rotations are allowed.)

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

color

The color in which to draw the overlay.

Remarks

Drawing is done in the device context associated to the desired window. The current pen is used.

EMatcher.DrawPositions

Draws a graphical representation of all occurrences of the pattern in the image, using a rectangle and possibly a small line segment in the upper-left corner.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void DrawPositions(
    IntPtr graphicContext,
    bool bCorner,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void DrawPositions(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    bool bCorner,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void DrawPositions(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    bool bCorner,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

bCorner

TRUE if the corner mark is to be drawn. **FALSE** by default. (This mark is useful when large rotations are allowed.)

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

color

The color in which to draw the overlay.

Remarks

Drawing is done in the device context associated to the desired window. The current pen is used for all occurrences (to vary the colors, draw the objects separately using the [EMatcher::DrawPosition](#) method instead).

EMatcher.DrawPositionsWithCurrentPen

Draws a graphical representation of all occurrences of the pattern in the image, using a rectangle and possibly a small line segment in the upper-left corner.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void DrawPositionsWithCurrentPen(  
    IntPtr graphicContext,  
    bool bCorner,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

bCorner

TRUE if the corner mark is to be drawn. **FALSE** by default. (This mark is useful when large rotations are allowed.)

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

Remarks

Drawing is done in the device context associated to the desired window. The current pen is used for all occurrences (to vary the colors, draw the objects separately using the [EMatcher::DrawPosition](#) method instead).

EMatcher.DrawPositionWithCurrentPen

Draws a graphical representation of a given occurrence of the pattern in the image, using a rectangle and possibly a small line segment in the upper-left corner.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawPositionWithCurrentPen (
    IntPtr graphicContext,
    uint index,
    bool bCorner,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

index

Occurrence index, in range **0..NumPositions-1**.

bCorner

TRUE if the corner mark is to be drawn. **FALSE** by default. (This mark is useful when large rotations are allowed.)

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

Remarks

Drawing is done in the device context associated to the desired window. The current pen is used.

EMatcher.EMatcher

Constructs a matching context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EMatcher(
)
void EMatcher(
    uint maxNumDOF
)
void EMatcher(
    Euresys.Open_eVision_2_10.EMatcher other
)
```

Parameters

maxNumDOF

Maximum number of degrees of freedom (this number must be comprised between **2** and **5**).

other

Another [EMatcher](#) object to be copied in the new [EMatcher](#) object.

Remarks

With the default constructor (no argument), all parameters are initialized to their respective default values. The copy constructor constructs a matching context based on a pre-existing [EMatcher](#) object. The last constructor constructs a matching context with a specified number of degrees of freedom. **maximumNumberOfDegreesOfFreedom** is the maximum number of degrees of freedom that the [EMatcher](#) object being constructed will be allowed to use during its life. All other parameters are initialized to their respective default values. The degrees of freedom for a matching operation are the *translation* (2 modes), the *rotation* (1 mode), the *isotropic scaling* (1 mode) and the *anisotropic scaling* (1 more mode). There is no way to modify this parameter for an existing [EMatcher](#) object. The default value for **maximumNumberOfDegreesOfFreedom** is **5**, that is the maximum value. The minimum value for the number of degrees of freedom is **2**, in order to allow, at least, the pattern translation.

EMatcher.FilteringMode

Filtering mode.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EFilteringMode FilteringMode
{ get; set; }
```

Remarks

To achieve acceptable time performance, EasyMatch works by sub-sampling the pattern in the early phases of the processing. The filtering mode parameter allows to select the pre-processing type applied to the image before the decimation: averaging or low-pass filtering. By default, this property is set to [Uniform](#), whereas the [LowPass](#) mode is indicated if the image presents sharp gray-level transitions.

EMatcher.FinalReduction

Index of the last reduction.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint FinalReduction
{ get; set; }
```

Remarks

The pattern matching process is comprised of a few passes (typically 4) during which the position accuracy is improved by a factor of 2 (for all degrees of freedom). This is called a *reduction*. By default, the computation continues until an accuracy of one pixel is obtained. To speed up the process, the last reduction passes can be dropped, so lowering the accuracy. Anyway, the interpolation mode can then still be used. By default, this property is set to **0** (pixel accuracy). Value **1** corresponds to 2-pixels accuracy, **2** to 4, and so on. The range of values that can be used for this property is **0** to **NumReductions-1**.

EMatcher.GetPixelDimensions

Gets the physical pixel dimensions.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetPixelDimensions (
    out float width,
    out float height
)
```

Parameters

width

Width of a pixel.

height

Height of a pixel.

EMatcher.GetPosition

Returns an [EMatchPosition](#) object containing the position coordinates.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EMatchPosition GetPosition(  
    uint index  
)
```

Parameters

index

0-based index to the desired position. The positions are ordered by decreasing score.

EMatcher.InitialMinScore

Minimum score applied as a selection criterion in the early stages of the matching process.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float InitialMinScore  
    { get; set; }
```

Remarks

When the search for matches starts, EasyMatch considers a set of candidate positions that it progressively refines. When they later appear to be bad candidates, they are rejected. Though it is the minimum score level that is used to reject bad matching positions at the final step of the matching process, the "initial minimum score" parameter is used to eliminate bad positions (whose score is not high enough) in the early stages of the matching processing. If the matching process is achieved in one step, only the minimum score parameter will be considered. By default, this property is set to **-1**.

EMatcher.Interpolate

Interpolation mode.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool Interpolate  
  
{ get; set; }
```

Remarks

By default, matching is done with a one-pixel precision for all degrees of freedom (translation, rotation and scaling). You can use an additional interpolation process to achieve sub-pixel accuracy. This generally leads to an improvement of the sub-pixel accuracy by a factor larger than 10. This is possible only when the found instances match closely the model. A score higher than 0.99 indicates that the instances are a close match of the model. In other words, the instance is considered to be more accurate when the score is higher. The added computational cost is low. By default, this property is set to **FALSE**.

EMatcher.IsotropicScale

Flag indicating whether isotropic (as opposed to anisotropic) scaling is used.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool IsotropicScale
{ get; }
```

Remarks

TRUE if isotropic (as opposed to anisotropic) scaling is used, i.e. when the scale factors in both the X and Y directions are equal.

EMatcher.LearnPattern

Learns a pattern to subsequently match in an image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void LearnPattern(
    Euresys.Open_eVision_2_10.EROIBW8 pattern
)
void LearnPattern(
    Euresys.Open_eVision_2_10.EROIC24 pattern
)
```

Parameters

pattern
Pattern to learn.

Remarks

The maximum size for a pattern is 1791x1791.

EMatcher.Load

Loads the [EMatcher](#). The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Load(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The serializer.

EMatcher.Match

Matches the pattern against an image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Match(
    Euresys.Open_eVision_2_10.EROIBW8 image
)

void Match(
    Euresys.Open_eVision_2_10.EROIC24 image
)
```

Parameters

image

Pointer to the image/ROI within which the pattern will be searched for.

Remarks

The matching results can be obtained by means of the [EMatcher::NumPositions](#) and [EMatcher::GetPosition](#) members.

EMatcher.MaxAngle

Maximum angle, in the current angle unit.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
float MaxAngle  
  
{ get; set; }
```

Remarks

The rotation of the pattern is allowed within the range $(-1 \leq \text{MinAngle} < \text{MaxAngle} \leq 1)$ revolution). By default, both remain **0**.

EMatcher.MaxInitialPositions

Maximum number of positions at the first stage of the matching process.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
uint MaxInitialPositions  
  
{ get; set; }
```

Remarks

When the search for matches starts, EasyMatch considers a set of candidate positions that it progressively refines. When they later appear to be bad candidates, they are rejected. Eventually, a maximum of [EMatcher::MaxPositions](#) is returned. In some circumstances, when the image contains features roughly similar to the pattern, these can confuse the matching process, resulting in false matches. To overcome this situation, increasing the number of initial positions will help. By default, this property is set to **0**, indicating that the value of [EMatcher::MaxPositions](#) should be used instead.

EMatcher.MaxPositions

Maximum number of positions.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
uint MaxPositions  
  
{ get; set; }
```

Remarks

Indicates how many matching positions have to be returned at a maximum. This number corresponds to the expected maximum number of occurrences of the pattern (it is sometimes advisable to use a few additional positions). By default, this property is set to **1**.

EMatcher.MaxScale

Maximum scale factor for isotropic scaling.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float MaxScale
```

```
{ get; set; }
```

Remarks

Two scaling modes are allowed: in isotropic mode, the scale factor is identical in all directions; in anisotropic mode, the scale factors differ in the horizontal and vertical directions. To select the appropriate mode, it suffices to call the relevant **Set** member. The scaling of the pattern is allowed within the range (**0.5 <= MinScale < MaxScale <= 2**). By default, both remain **1**. The same holds for anisotropic scale factors.

EMatcher.MaxScaleX

Maximum horizontal scale factor for anisotropic scaling.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float MaxScaleX
```

```
{ get; set; }
```

Remarks

Two scaling modes are allowed: in isotropic mode, the scale factor is identical in all directions; in anisotropic mode, the scale factors differ in the horizontal and vertical directions. To select the appropriate mode, it suffices to call the relevant **Set** member. The scaling of the pattern is allowed within the range (**0.5 <= MinScaleX < MaxScaleX <= 2**). By default, both remain **1**. The same holds for anisotropic scale factors.

EMatcher.MaxScaleY

Maximum vertical scale factor for anisotropic scaling.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float MaxScaleY
{ get; set; }
```

Remarks

Two scaling modes are allowed: in isotropic mode, the scale factor is identical in all directions; in anisotropic mode, the scale factors differ in the horizontal and vertical directions. To select the appropriate mode, it suffices to call the relevant **Set** member. The scaling of the pattern is allowed within the range (**0.5 <= MinScaleY < MaxScaleY <= 2**). By default, both remain **1**. The same holds for anisotropic scale factors.

EMatcher.MinAngle

Minimum angle, in the current angle unit.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float MinAngle
{ get; set; }
```

Remarks

The rotation of the pattern is allowed within the range (**-1 <= MinAngle < MaxAngle <= 1** revolution). By default, both remain **0**.

EMatcher.MinReducedArea

Minimum reduced area parameter.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint MinReducedArea
{ get; set; }
```

Remarks

To achieve acceptable time performance, EasyMatch works by under-sampling the pattern in the early phases of the processing. This property tells how many pixels of the pattern are kept, at a minimum. By default, this property is set to **64**, which is the right choice in most situations. Higher values are not recommended. Lower values can speed up the processing, but sometimes cause the matching process fail to find the best matches. To circumvent this problem, you can use guard positions, that is find more matches than expected and keep the good ones only.

Note. Changing this property invalidates any previous learning.

EMatcher.MinScale

Minimum scale factor for isotropic scaling.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float MinScale
{ get; set; }
```

Remarks

Two scaling modes are allowed: in isotropic mode, the scale factor is identical in all directions; in anisotropic mode, the scale factors differ in the horizontal and vertical directions. To select the appropriate mode, it suffices to call the relevant **Set** member. The scaling of the pattern is allowed within the range (**0.5 <= MinScale < MaxScale <= 2**). By default, both remain **1**. The same holds for anisotropic scale factors.

EMatcher.MinScaleX

Minimum horizontal scale factor for anisotropic scaling.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
float MinScaleX  
  
{ get; set; }
```

Remarks

Two scaling modes are allowed: in isotropic mode, the scale factor is identical in all directions; in anisotropic mode, the scale factors differ in the horizontal and vertical directions. To select the appropriate mode, it suffices to call the relevant **Set** member. The scaling of the pattern is allowed within the range (**0.5** <= **MinScaleX** < **MaxScaleX** <= **2**). By default, both remain **1**. The same holds for anisotropic scale factors.

EMatcher.MinScaleY

Minimum vertical scale factor for anisotropic scaling.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
float MinScaleY  
  
{ get; set; }
```

Remarks

Two scaling modes are allowed: in isotropic mode, the scale factor is identical in all directions; in anisotropic mode, the scale factors differ in the horizontal and vertical directions. To select the appropriate mode, it suffices to call the relevant **Set** member. The scaling of the pattern is allowed within the range (**0.5** <= **MinScaleY** < **MaxScaleY** <= **2**). By default, both remain **1**. The same holds for anisotropic scale factors.

EMatcher.MinScore

Minimum score.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
float MinScore  
  
    { get; set; }
```

Remarks

This property indicates what score a match must reach to be considered as good, and to be returned in the list of positions; this selection criterion is applied at the final stage of the matching process. One good way to select the appropriate [EMatcher::MinScore](#) in a given context is to set it to **-1** (any match will be retained), set [EMatcher::MaxPositions](#) to more than needed, and examine the returned scores after a matching. By default, this property is set to **-1**.

EMatcher.NumPositions

Number of good matches found, as defined by [EMatcher::MinScore](#) and [EMatcher::MaxPositions](#) properties.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
uint NumPositions  
  
    { get; }
```

EMatcher.NumReductions

Number of reduction steps used in the matching process.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
uint NumReductions
{ get; }
```

Remarks

These depend on the actual pattern size, and on the **MinReducedArea** property. The **FinalReduction** property, used to speed up matching when coarse location is sufficient, must be set in range **0..NumReductions-1**.

EMatcher.operator=

Copies all the data from another [EMatcher](#) object into the current [EMatcher](#) object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EMatcher operator=(
    Euresys.Open_eVision_2_10.EMatcher other
)
```

Parameters

other
[EMatcher](#) object to be copied

EMatcher.PatternHeight

Learnt pattern height.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int PatternHeight  
  
{ get; }
```

EMatcher.PatternLearnt

Returns **TRUE** after a learning operation has been successfully performed, indicating that the [EMatcher](#) object is ready for matching, and **FALSE** otherwise.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool PatternLearnt  
  
{ get; }
```

EMatcher.PatternType

Pixel type of the learnt pattern.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EImageType PatternType  
  
{ get; }
```

EMatcher.PatternWidth

Learnt pattern width.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int PatternWidth
    { get; }
```

EMatcher.Positions

Returns a vector of [EMatchPosition](#) objects, each containing the position coordinates and other matching results.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EMatchPosition[] Positions
    { get; }
```

EMatcher.Save

Saves the [EMatcher](#). The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The serializer.

EMatcher.ScaleStep

Current value of scale step.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float ScaleStep
{ get; }
```

EMatcher.ScaleXStep

Current value of scale X step.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float ScaleXStep
{ get; }
```

EMatcher.ScaleYStep

Current value of scale Y step.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ScaleYStep  
    { get; }
```

EMatcher.SetExtension

Sets the extension of the matching ROI, i.e. puts the horizontal and vertical distances, in pixels, that the found pattern occurrences may fall outside the matching ROI. Such occurrences partially outside the ROI have their score corrected by the ratio between the pattern area outside the ROI and the pattern area inside.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void SetExtension(  
    int n32ExtensionX,  
    int n32ExtensionY  
)
```

Parameters

n32ExtensionX

extension outside the matching ROI along its Width, in pixels.

n32ExtensionY

extension outside the matching ROI along its Height, in pixels.

EMatcher.SetPixelDimensions

Sets the physical pixel dimensions.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetPixelDimensions (
    float width,
    float height
)
```

Parameters

width

Width of a pixel.

height

Height of a pixel.

Remarks

When an image has been acquired in such a way that the pixels are "non-square" –the physical width and height of the area covered by a pixel are unequal– a form of anisotropy results. When rotated, the objects become skewed; rectangles become parallelograms. In such a situation, the pixel aspect ratio must be known to compensate it during matching. The aspect ratio is given by specifying the true width and height of a pixel. The specification of the pixel dimensions is only useful when rotation is used. Only the aspect ratio matters, so that relative width and height values can be given. By default, the pixel width and height are set to **1.0**.

EMatcher.Version

Version number of the [EMatcher](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
static uint Version
{ get; }
```

4.110. EMatrixCode Class

Holds all the information regarding a single Data Matrix code: its decoded string, its grading, its errors,...

Namespace: Euresys.Open_eVision_2_10

Properties

Angle	MatrixCode angle.
AxialNonUniformity	Measured axial non-uniformity value (1.0 to 0 scale), as defined by the AIM standard, after a read operation, provided that the print quality assessment has been enabled.
AxialNonUniformityGrade	Measured axial non-uniformity grading (4 to 0 scale), as defined by the AIM standard, after a read operation, provided that the print quality assessment has been enabled.
CellDefects	Measured cell defects value, as defined by the Semi T10 standard, after a read operation, provided that the print quality assessment has been enabled.
Center	EMatrixCode center.

Contrast

Measured symbol contrast value, as defined by the AIM standard, after a read operation, provided that the print quality assessment has been enabled.

ContrastGrade

Measured symbol contrast grading (**4** to **0** scale), as defined by the AIM standard, after a read operation, provided that the print quality assessment has been enabled.

ContrastType

Symbol contrast type, as defined by [EMatrixCodeContrastMode](#).

DataMatrixCellHeight

Measured data matrix cell height value, as defined by the Semi T10 standard, after a read operation, provided that the print quality assessment has been enabled.

DataMatrixCellWidth

Measured data matrix cell width value, as defined by the Semi T10 standard, after a read operation, provided that the print quality assessment has been enabled.

DecodedString

Decoded Data Matrix symbol string.

Family

ECC symbol family, as defined by [EFamily](#).

FinderPatternDefects

Measured finder pattern defects value, as defined by the Semi T10 standard, after a read operation, provided that the print quality assessment has been enabled.

Flipping	Symbol flipping type, as defined by EFlipping .
Found	TRUE if a EMatrixCode object has been found in the ROI supplied to EMatrixCodeReader::Read , even if it could not be successfully decoded.
HorizontalMarkGrowth	Measured horizontal mark growth value, as defined by the Semi T10 standard, after a read operation, provided that the print quality assessment has been enabled.
HorizontalMarkMisplacement	Measured horizontal mark misplacement value, as defined by the Semi T10 standard, after a read operation, provided that the print quality assessment has been enabled.
Iso15415GradingParameters	ISO/IEC 15415 grading parameters
Iso29158GradingParameters	ISO/IEC 29158 grading parameters
LocationThreshold	Absolute threshold (0 to 255 scale) that was used during location of the symbol.
LogicalSize	Symbol logical size, as defined by ELogicalSize .
LogicalSizeHeight	For a logical size of S1xS2, LogicalSizeHeight is the integer S1.
LogicalSizeWidth	For a logical size of S1xS2, LogicalSizeWidth is the integer S2.

MeasuredPrintGrowth

Raw, un-normalized print quality parameter (**1.0** to **0** scale), after a read operation, provided that the print quality assessment has been enabled.

NumErrors

Number of errors that were detected and corrected while decoding the symbol.

OverallGrade

Overall symbol grading (**4** to **0** scale), as defined by the AIM standard, after a read operation, provided that the print quality assessment has been enabled.

PrintGrowth

Normalized measured print growth value, as defined by the AIM standard, after a read operation, provided that the print quality assessment has been enabled.

PrintGrowthGrade

Measured print growth grading (**4** to **0** scale), as defined by the AIM standard, after a read operation, provided that the print quality assessment has been enabled.

ReadingThreshold

Absolute threshold (**0** to **255** scale) that was used during reading of the symbol.

SemiT10GradingParameters

Semi T10-0701 grading parameters

SymbolContrastSNR

Measured symbol contrast signal to noise ratio value, as defined by the Semi T10 standard, after a read operation, provided that the print quality assessment has been enabled.

UnusedErrorCorrection

Measured unused error correction value (**1.0** to **0** scale), as defined by the AIM standard, after a read operation, provided that the print quality assessment has been enabled.

UnusedErrorCorrectionGrade

Measured unused error correction grading (**4** to **0** scale), as defined by the AIM standard, after a read operation, provided that the print quality assessment has been enabled.

VerticalMarkGrowth

Measured vertical mark growth value, as defined by the Semi T10 standard, after a read operation, provided that the print quality assessment has been enabled.

VerticalMarkMisplacement

M Measured vertical mark misplacement value, as defined by the Semi T10 standard, after a read operation, provided that the print quality assessment has been enabled.

thods

Draw

Draws the EMatrixCode corner points and symbol finder pattern (when the symbol size has been correctly detected).

DrawErrors

Draws all symbol finder pattern cells where errors were detected and corrected.

DrawErrorsWithCurrentPen

Draws the detected errors.

DrawWithCurrentPen

Draws the EMatrixCode corner points and symbol finder pattern (when the symbol size has been correctly detected).

EMatrixCode

Constructs a EMatrixCode context.

GetCorner

Gets a matrix code corner.

GetDecodedDataElement

Gets a character value of the matrix code.

IsGS1

TRUE if the decoded string uses the GS1 standard

Load

Saves a [EMatrixCode](#). The given ESerializer must have been created for writing.

operator=

Copies all the data from another EMatrixCode object into the current EMatrixCode object

Save

Loads a [EMatrixCode](#). The given ESerializer must have been created for reading.

SetCorner

⌈ Sets a matrix code corner.

M

atrixCode.Angle

MatrixCode angle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Angle  
  
    { get; }
```

Remarks

EMatrixCode.AxialNonUniformity

Measured axial non-uniformity value (**1.0** to **0** scale), as defined by the AIM standard, after a read operation, provided that the print quality assessment has been enabled.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float AxialNonUniformity  
  
    { get; }
```

EMatrixCode.AxialNonUniformityGrade

Measured axial non-uniformity grading (**4** to **0** scale), as defined by the AIM standard, after a read operation, provided that the print quality assessment has been enabled.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int AxialNonUniformityGrade  
  
    { get; }
```

EMatrixCode.CellDefects

Measured cell defects value, as defined by the Semi T10 standard, after a read operation, provided that the print quality assessment has been enabled.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float CellDefects  
  
    { get; }
```

Remarks

Read-only.

EMatrixCode.Center

EMatrixCode center.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint Center  
  
    { get; }
```

EMatrixCode.Contrast

Measured symbol contrast value, as defined by the AIM standard, after a read operation, provided that the print quality assessment has been enabled.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
float Contrast  
  
{ get; }
```

Remarks

This property is computed as the difference of the reference gray levels over their arithmetic average. Values range between **0** and **1.0**.

EMatrixCode.ContrastGrade

Measured symbol contrast grading (**4** to **0** scale), as defined by the AIM standard, after a read operation, provided that the print quality assessment has been enabled.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
int ContrastGrade  
  
{ get; }
```

EMatrixCode.ContrastType

Symbol contrast type, as defined by [EMatrixCodeContrastMode](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EMatrixCodeContrastMode ContrastType  
{ get; }
```

EMatrixCode.DataMatrixCellHeight

Measured data matrix cell height value, as defined by the Semi T10 standard, after a read operation, provided that the print quality assessment has been enabled.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float DataMatrixCellHeight  
{ get; }
```

Remarks

Read-only.

EMatrixCode.DataMatrixCellWidth

Measured data matrix cell width value, as defined by the Semi T10 standard, after a read operation, provided that the print quality assessment has been enabled.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float DataMatrixCellWidth  
{ get; }
```


Remarks

Read-only.

EMatrixCode.DecodedString

Decoded Data Matrix symbol string.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
string DecodedString  
    { get; }
```

EMatrixCode.Draw

Draws the EMatrixCode corner points and symbol finder pattern (when the symbol size has been correctly detected).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Draw(  
    IntPtr graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

```

Parameters

graphicContext

Handle of the device context on which to draw.

zoomX

Horizontal zooming factor. A value greater than **1** means zoom in. By default, **TRUE** scale is used.

zoomY

Vertical zooming factor. A value greater than **1** means zoom in. If set to **0**, the default value, the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor, in pixels. By default, no panning occurs.

panY

Vertical panning factor, in pixels. By default, no panning occurs.

color

The color in which to draw the overlay.

Remarks

The reference corner has a bold cross marking.

EMatrixCode.DrawErrors

Draws all symbol finder pattern cells where errors were detected and corrected.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void DrawErrors(
    IntPtr graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void DrawErrors(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void DrawErrors(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

zoomX

Horizontal zooming factor. A value greater than **1** means zoom in. By default, **TRUE** scale is used.

zoomY

Vertical zooming factor. A value greater than **1** means zoom in. If set to **0**, the default value, the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor, in pixels. By default, no panning occurs.

panY

Vertical panning factor, in pixels. By default, no panning occurs.

color

The color in which to draw the overlay.

Remarks

This member is intended to be called in conjunction with [EMatrixCode::Draw](#).

EMatrixCode.DrawErrorsWithCurrentPen

Draws the detected errors.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawErrorsWithCurrentPen (
    IntPtr graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

Parameters

graphicContext

-

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

EMatrixCode.DrawWithCurrentPen

Draws the EMatrixCode corner points and symbol finder pattern (when the symbol size has been correctly detected).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawWithCurrentPen(
    IntPtr graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

zoomX

Horizontal zooming factor. A value greater than **1** means zoom in. By default, **TRUE** scale is used.

zoomY

Vertical zooming factor. A value greater than **1** means zoom in. If set to **0**, the default value, the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor, in pixels. By default, no panning occurs.

panY

Vertical panning factor, in pixels. By default, no panning occurs.

Remarks

The reference corner has a bold cross marking.

EMatrixCode.EMatrixCode

Constructs a EMatrixCode context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EMatrixCode(
)

void EMatrixCode(
    Euresys.Open_eVision_2_10.EMatrixCode other
)
```

Parameters

other

Another EMatrixCode object to be copied in the new EMatrixCode object.

Remarks

The default constructor constructs an uninitialized EMatrixCode object. All properties are initialized to their respective default values. The copy constructor constructs a EMatrixCode context based on a pre-existing EMatrixCode object. All properties and internal data are copied.

EMatrixCode.Family

ECC symbol family, as defined by [EFamily](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EFamily Family
{ get; }
```

EMatrixCode.FinderPatternDefects

Measured finder pattern defects value, as defined by the Semi T10 standard, after a read operation, provided that the print quality assessment has been enabled.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float FinderPatternDefects
{ get; }
```

Remarks

Read-only.

EMatrixCode.Flipping

Symbol flipping type, as defined by [EFlipping](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EFlipping Flipping
{ get; }
```

EMatrixCode.Found

TRUE if a EMatrixCode object has been found in the ROI supplied to [EMatrixCodeReader::Read](#), even if it could not be successfully decoded.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool Found
{ get; }
```

Remarks

If this property is **FALSE**, it is still possible that an unlocalized matrix code exists in the image. However, if **Found** is **FALSE**, no other property should be read.

EMatrixCode.GetCorner

Gets a matrix code corner.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint GetCorner(
    int index
)
```

Parameters

index

Index of the matrix code corner.

EMatrixCode.GetDecodedDataElement

Gets a character value of the matrix code.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
byte GetDecodedDataElement(
    int index
)
```


Parameters

index

Index of the character value.

Remarks

This property makes it possible to see information not coded as ASCII characters.

EMatrixCode.HorizontalMarkGrowth

Measured horizontal mark growth value, as defined by the Semi T10 standard, after a read operation, provided that the print quality assessment has been enabled.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float HorizontalMarkGrowth  
  
    { get; }
```

Remarks

Read-only.

EMatrixCode.HorizontalMarkMisplacement

Measured horizontal mark misplacement value, as defined by the Semi T10 standard, after a read operation, provided that the print quality assessment has been enabled.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float HorizontalMarkMisplacement
```

```
{ get; }
```

Remarks

Read-only.

EMatrixCode.IsGS1

TRUE if the decoded string uses the GS1 standard

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool IsGS1 (  
)
```

EMatrixCode.Iso15415GradingParameters

ISO/IEC 15415 grading parameters

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EMatrixCodeIso15415GradingParameters Iso15415GradingParameters  
{ get; }
```

Remarks

Read-only.

EMatrixCode.Iso29158GradingParameters

ISO/IEC 29158 grading parameters

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EMatrixCodeIso29158GradingParameters Iso29158GradingParameters  
    { get; }
```

Remarks

Read-only.

EMatrixCode.Load

Saves a [EMatrixCode](#). The given ESerializer must have been created for writing.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Load(  
    Euresys.Open_eVision_2_10.ESerializer serializer  
)
```

Parameters

serializer

The serializer.

EMatrixCode.LocationThreshold

Absolute threshold (**0** to **255** scale) that was used during location of the symbol.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int LocationThreshold
{ get; }
```

EMatrixCode.LogicalSize

Symbol logical size, as defined by [ELogicalSize](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ELogicalSize LogicalSize
{ get; }
```

EMatrixCode.LogicalSizeHeight

For a logical size of S1xS2, **LogicalSizeHeight** is the integer S1.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int LogicalSizeHeight
{ get; }
```

EMatrixCode.LogicalSizeWidth

For a logical size of S1xS2, **LogicalSizeWidth** is the integer S2.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int LogicalSizeWidth
{ get; }
```

EMatrixCode.MeasuredPrintGrowth

Raw, un-normalized print quality parameter (**1.0** to **0** scale), after a read operation, provided that the print quality assessment has been enabled.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float MeasuredPrintGrowth
{ get; }
```

Remarks

This property is computed as the measured area of the active cells (same color as the finder pattern) over the area of ideal square cells, having sides equal to the pitches (such cells perfectly tile the symbol). Its value typically is on the order of **1.0**, regardless the symbol size.

EMatrixCode.NumErrors

Number of errors that were detected and corrected while decoding the symbol.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int NumErrors
{ get; }
```

Remarks

Such errors may be due to symbol degradation by scratches, blur, non-uniform illumination or slight changes in size.

EMatrixCode.operator=

Copies all the data from another EMatrixCode object into the current EMatrixCode object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EMatrixCode operator=(
    Euresys.Open_eVision_2_10.EMatrixCode other
)
```

Parameters

other

EMatrixCode object to be copied

EMatrixCode.OverallGrade

Overall symbol grading (**4** to **0** scale), as defined by the AIM standard, after a read operation, provided that the print quality assessment has been enabled.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int OverallGrade  
    { get; }
```

EMatrixCode.PrintGrowth

Normalized measured print growth value, as defined by the AIM standard, after a read operation, provided that the print quality assessment has been enabled.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float PrintGrowth  
    { get; }
```

Remarks

The use of this property is a bit tricky: first a raw measure of the print growth is provided as [EMatrixCode::MeasuredPrintGrowth](#). Then, the measurement is normalized from the [EMatrixCodeReader::MinimumPrintGrowth](#) / [EMatrixCodeReader::MaximumPrintGrowth](#) / [EMatrixCodeReader::NominalPrintGrowth](#) properties of the [EMatrixCodeReader](#) instance, which must be provided by the user. The normalized [EMatrixCode::PrintGrowth](#) ranges around **0**.

EMatrixCode.PrintGrowthGrade

Measured print growth grading (**4** to **0** scale), as defined by the AIM standard, after a read operation, provided that the print quality assessment has been enabled.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int PrintGrowthGrade  
  
    { get; }
```

Remarks

Read-only.

EMatrixCode.ReadingThreshold

Absolute threshold (**0** to **255** scale) that was used during reading of the symbol.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int ReadingThreshold  
  
    { get; }
```

Remarks

Read-only.

EMatrixCode.Save

Loads a [EMatrixCode](#). The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is read from.

EMatrixCode.SemiT10GradingParameters

Semi T10-0701 grading parameters

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EMatrixCodeSemiT10GradingParameters SemiT10GradingParameters
{ get; }
```

Remarks

Read-only.

EMatrixCode.SetCorner

Sets a matrix code corner.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetCorner(
    int index,
    Euresys.Open_eVision_2_10.EPoint corner
)
```

Parameters

index

Index of the matrix code corner.

corner

New corner.

EMatrixCode.SymbolContrastSNR

Measured symbol contrast signal to noise ratio value, as defined by the Semi T10 standard, after a read operation, provided that the print quality assessment has been enabled.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float SymbolContrastSNR
{ get; }
```

Remarks

Read-only.

EMatrixCode.UnusedErrorCorrection

Measured unused error correction value (**1.0** to **0** scale), as defined by the AIM standard, after a read operation, provided that the print quality assessment has been enabled.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float UnusedErrorCorrection  
  
    { get; }
```

Remarks

The [EMatrixCode::UnusedErrorCorrection](#) property takes into account the number of redundant bits used for error correction only; no erasure nor error detection bits are considered.

EMatrixCode.UnusedErrorCorrectionGrade

Measured unused error correction grading (**4** to **0** scale), as defined by the AIM standard, after a read operation, provided that the print quality assessment has been enabled.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int UnusedErrorCorrectionGrade  
  
    { get; }
```

Remarks

Read-only.

EMatrixCode.VerticalMarkGrowth

Measured vertical mark growth value, as defined by the Semi T10 standard, after a read operation, provided that the print quality assessment has been enabled.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float VerticalMarkGrowth  
  
    { get; }
```

Remarks

Read-only.

EMatrixCode.VerticalMarkMisplacement

Measured vertical mark misplacement value, as defined by the Semi T10 standard, after a read operation, provided that the print quality assessment has been enabled.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float VerticalMarkMisplacement  
  
    { get; }
```

Remarks

Read-only.

4.111. EMatrixCode Class

Holds all the information regarding a single Data Matrix code: its decoded string, its grading, its errors and more.

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

Properties

DecodedString	Decoded Data Matrix symbol string.
ECC000Family	Retrieves the ECC000 Family for non-ECC200 Matrix Codes.
Errors	Retrieves the position of the errors detected in the Data Matrix symbol.
IsECC200	Indicates if the Data Matrix is ECC200 or not.
Iso15415GradingParameters	ISO/IEC 15415 grading parameters
Iso29158GradingParameters	ISO/IEC TR 29158 grading parameters
Position	Position of the Data Matrix
SemiT10GradingParameters	Semi T10-0701 grading parameters

SymbolHeight

Data Matrix Symbol Height (Number of cells in the vertical direction)

SymbolWidth

M Data Matrix Symbol Width (Number of cells in the horizontal direction)

e

thods

DrawErrors

Draws the detected errors.

DrawGrid

Draws the detected Data Matrix grid.

DrawPosition

Draws the Data Matrix Position. This includes the outer edges of the code as well as the timing patterns.

EMatrixCode

Creates an [EMatrixCode](#) object.

GetCellPosition

Position of a cell of the Data Matrix

IsGS1

TRUE if the decoded string uses the GS1 standard

operator=

Assignment operator

EMatrixCode.DecodedString

Decoded Data Matrix symbol string.

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]
string DecodedString
{ get; }
```

EMatrixCode.DrawErrors

Draws the detected errors.

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]
void DrawErrors(
    IntPtr hDC,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

Parameters

hDC

Handle of the device context on which to draw.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

EMatrixCode.DrawGrid

Draws the detected Data Matrix grid.

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]
void DrawGrid(
    IntPtr hDC,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

Parameters

hDC

Handle of the device context on which to draw.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

EMatrixCode.DrawPosition

Draws the Data Matrix Position. This includes the outer edges of the code as well as the timing patterns.

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]
void DrawPosition(
    IntPtr hDC,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

Parameters

hDC

Handle of the device context on which to draw.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

EMatrixCode.ECC000Family

Retrieves the ECC000 Family for non-ECC200 Matrix Codes.

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]
Euresys.Open_eVision_2_10.EasyMatrixCode2.ECC000Family ECC000Family
{ get; }
```

EMatrixCode.EMatrixCode

Creates an [EMatrixCode](#) object.

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]
void EMatrixCode(
)
void EMatrixCode(
    Euresys.Open_eVision_2_10.EasyMatrixCode2.EMatrixCode other
)
```

Parameters

other

The reference [EMatrixCode](#) instance to copy this one from.

EMatrixCode.Errors

Retrieves the position of the errors detected in the Data Matrix symbol.

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]
Euresys.Open_eVision_2_10.EMatrixPosition[] Errors
{ get; }
```

EMatrixCode.GetCellPosition

Position of a cell of the Data Matrix

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]
Euresys.Open_eVision_2_10.EQuadrangle GetCellPosition(
    int x,
    int y
)
Euresys.Open_eVision_2_10.EQuadrangle GetCellPosition(
    Euresys.Open_eVision_2_10.EMatrixPosition position
)
```

Parameters

x
The horizontal index of the cell.

y
The vertical index of the cell.

position
The position of the cell in the code.

EMatrixCode.IsECC200

Indicates if the Data Matrix is ECC200 or not.

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]  
bool IsECC200  
  
    { get; }
```

EMatrixCode.IsGS1

TRUE if the decoded string uses the GS1 standard

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]  
bool IsGS1 (  
    )
```

EMatrixCode.Iso15415GradingParameters

ISO/IEC 15415 grading parameters

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]  
Euresys.Open_eVision_2_10.EMatrixCodeIso15415GradingParameters Iso15415GradingParameters  
  
    { get; }
```

EMatrixCode.Iso29158GradingParameters

ISO/IEC TR 29158 grading parameters

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]
Euresys.Open_eVision_2_10.EMatrixCodeIso29158GradingParameters Iso29158GradingParameters
{ get; }
```

EMatrixCode.operator=

Assignment operator

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]
Euresys.Open_eVision_2_10.EasyMatrixCode2.EMatrixCode operator=(
    Euresys.Open_eVision_2_10.EasyMatrixCode2.EMatrixCode other
)
```

Parameters

other

The [EMatrixCode](#) instance to assign.

EMatrixCode.Position

Position of the Data Matrix

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]  
Euresys.Open_eVision_2_10.EQuadrangle Position  
    { get; }
```

EMatrixCode.SemiT10GradingParameters

Semi T10-0701 grading parameters

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]  
Euresys.Open_eVision_2_10.EMatrixCodeSemiT10GradingParameters SemiT10GradingParameters  
    { get; }
```

EMatrixCode.SymbolHeight

Data Matrix Symbol Height (Number of cells in the vertical direction)

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]  
int SymbolHeight  
    { get; }
```

EMatrixCode.SymbolWidth

Data Matrix Symbol Width (Number of cells in the horizontal direction)

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]
int SymbolWidth
{ get; }
```

4.112. EMatrixCodeReader Class

A [EMatrixCodeReader](#) instance is a tool that processes an ROI and returns a [EMatrixCode](#) instance.

Remarks

Namespace: Euresys.Open_eVision_2_10

Properties

ComputeGrading

Allows to choose whether the grading properties of the [EMatrixCode](#) object will be computed by the [EMatrixCodeReader::Reset](#), [EMatrixCodeReader::Read](#) or [EMatrixCodeReader::LearnMore](#) methods.

MaxHeightWidthRatio

Maximum value for a Data Matrix aspect ratio

MaximumPrintGrowth

Maximum reference value in use for normalization of the **PrintGrowth** quality indicator.

MinimumPrintGrowth

Minimum reference value in use for normalization of the **PrintGrowth** quality indicator.

NominalPrintGrowth

Nominal reference value in use for normalization of the **PrintGrowth** quality indicator.

SearchParams

Parameter space that the algorithm uses to read a Data Matrix code from an ROI.

TimeOut

M Time-out for the [EMatrixCodeReader::Learn](#), [EMatrixCodeReader::LearnMore](#) and **Read** methods.

e

thods

EMatrixCodeReader

Default constructor for EMatrixCodeReader objects.

GetLearnMaskElement

Allows to know which decoded parameters are learnt when the [EMatrixCodeReader::Learn](#) or [EMatrixCodeReader::LearnMore](#) methods are called.

Learn

Tries to locate, decode and read the Data Matrix code in the given ROI.

[LearnMore](#)

Tries to locate, decode and read the Data Matrix code in the given ROI.

[Load](#)

Saves a [EMatrixCodeReader](#). The given ESerializer must have been created for writing.

[Read](#)

Tries to locate, decode and read the Data Matrix code in the given ROI.

[Reset](#)

Resets the parameter search space to its default: the full range of all parameters.

[Save](#)

Loads a [EMatrixCodeReader](#). The given ESerializer must have been created for reading.

[SetIso29158CalibrationParameters](#)

Sets ISO/IEC 29158 calibration paramters

[SetLearnMaskElement](#)

EMatrixCodeReader.ComputeGrading

Allows to choose which decoded parameters are learnt when the

Allows to choose whether the grading properties of the

[EMatrixCode](#) object will be computed by the [EMatrixCodeReader::Reset](#), [EMatrixCodeReader::Read](#) or [EMatrixCodeReader::LearnMore](#) methods.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool ComputeGrading
    { get; set; }
```

Remarks

Default: **FALSE**.

EMatrixCodeReader.EMatrixCodeReader

Default constructor for EMatrixCodeReader objects.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EMatrixCodeReader (
)
void EMatrixCodeReader (
    Euresys.Open_eVision_2_10.EMatrixCodeReader other
)
```

Parameters

other

-

EMatrixCodeReader.GetLearnMaskElement

Allows to know which decoded parameters are learnt when the [EMatrixCodeReader::Learn](#) or [EMatrixCodeReader::LearnMore](#) methods are called.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool GetLearnMaskElement(
    Euresys.Open_eVision_2_10.ELearnParam index
)
```

Parameters

index

Parameter identifier, as defined in [ELearnParam](#)

EMatrixCodeReader.Learn

Tries to locate, decode and read the Data Matrix code in the given ROI.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EMatrixCode Learn(
    Euresys.Open_eVision_2_10.EROIBW8 roi
)
```

Parameters

roi

ROI in which the Data Matrix has to be found.

Remarks

If successful, it adds the parameters of the Data Matrix code found into the internal learning database. The decoding results can be found in the returned [EMatrixCode](#) object. The addition of the parameters of the Data Matrix code found into the internal learning database means that subsequent **Read** operations will be faster, but can only be performed on similar Data Matrix codes/conditions. Only the parameters that are tagged for learning are remembered for the subsequent [EMatrixCodeReader::Read](#) operations (see [EMatrixCodeReader::SetLearnMaskElement](#)). See the [EMatrixCode::Found](#) property for information about the outcome of the **Learn** process.

EMatrixCodeReader.LearnMore

Tries to locate, decode and read the Data Matrix code in the given ROI.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EMatrixCode LearnMore(  
    Euresys.Open_eVision_2_10.EROIBW8 roi  
)
```

Parameters

roi

ROI in which the Data Matrix has to be found.

Remarks

If successful, it cumulates the parameters of the Data Matrix code found with those already present in the internal learning database. The decoding results can be found in the returned [EMatrixCode](#) object. The cumulation of the parameters of the Data Matrix code found with those already present in the internal learning database means that subsequent **Read** operations will be faster, but can only be performed on similar Data Matrix codes/conditions. Only the parameters that are tagged for learning are remembered for the subsequent [EMatrixCodeReader::Read](#) operations (see [EMatrixCodeReader::SetLearnMaskElement](#)). See the [EMatrixCode::Found](#) property for information about the outcome of the **LearnMore** process.

EMatrixCodeReader.Load

Saves a [EMatrixCodeReader](#). The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void Load(  
    Euresys.Open_eVision_2_10.ESerializer serializer  
)
```

Parameters

serializer

The serializer.

EMatrixCodeReader.MaxHeightWidthRatio

Maximum value for a Data Matrix aspect ratio

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float MaxHeightWidthRatio  
  
{ get; set; }
```

Remarks

This property allows controlling what kind of objects are considered as potential MatrixCode instances in the image. When objects are found in the image, only those where the bounding box has an aspect ratio smaller than this value are taken into account for digitization and decoding. The default value is 3.8, and should be adjusted if the MatrixCode cells in your image are non-square, or if your matrix code uses a very non-square symbology such as 32x8. The supplied value must lie between 0.0 and 5.0.

EMatrixCodeReader.MaximumPrintGrowth

Maximum reference value in use for normalization of the **PrintGrowth** quality indicator.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float MaximumPrintGrowth
{ get; set; }
```

Remarks

Default: **2.0**. After the standard, parameter **PrintGrowth** must be computed as normalized value related to three references: minimum, nominal and maximum values have to be provided. Parameter **MeasuredPrintGrowth** is the raw, un-normalized print quality parameter. It is computed as the measured area of the active cells over the area of ideal square cells, having sides equal to the pitches (such cells perfectly tile the symbol). Its value typically is on the order of unity. The **PrintGrowth** is derived from the **MeasuredPrintGrowth** by means of the three normalization parameters.

EMatrixCodeReader.MinimumPrintGrowth

Minimum reference value in use for normalization of the **PrintGrowth** quality indicator.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float MinimumPrintGrowth
{ get; set; }
```

Remarks

Default: **0.0**. After the standard, parameter **PrintGrowth** must be computed as normalized value related to three references: minimum, nominal and maximum values have to be provided. Parameter **MeasuredPrintGrowth** is the raw, un-normalized print quality parameter. It is computed as the measured area of the active cells over the area of ideal square cells, having sides equal to the pitches (such cells perfectly tile the symbol). Its value typically is on the order of unity. The **PrintGrowth** is derived from the **MeasuredPrintGrowth** by means of the three normalization parameters.

EMatrixCodeReader.NominalPrintGrowth

Nominal reference value in use for normalization of the **PrintGrowth** quality indicator.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
float NominalPrintGrowth  
  
    { get; set; }
```

Remarks

Default: **1.0**. After the standard, parameter **PrintGrowth** must be computed as normalized value related to three references: minimum, nominal and maximum values have to be provided. Parameter **MeasuredPrintGrowth** is the raw, un-normalized print quality parameter. It is computed as the measured area of the active cells over the area of ideal square cells, having sides equal to the pitches (such cells perfectly tile the symbol). Its value typically is on the order of unity. The **PrintGrowth** is derived from the **MeasuredPrintGrowth** by means of the three normalization parameters.

EMatrixCodeReader.Read

Tries to locate, decode and read the Data Matrix code in the given ROI.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
Euresys.Open_eVision_2_10.EMatrixCode Read(  
    Euresys.Open_eVision_2_10.EROIBW8 roi  
)
```

Parameters

roi

ROI in which the Data Matrix has to be found.

Remarks

The decoding results can be found in the returned [EMatrixCode](#) object. See the [EMatrixCode::Found](#) property for information about the outcome of the **Read** process.

Note. This function throws an exception if the matrix code in the given ROI can not be read.

EMatrixCodeReader.Reset

Resets the parameter search space to its default: the full range of all parameters.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Reset(  
    )
```

Remarks

This does not modify the learning mask.

EMatrixCodeReader.Save

Loads a [EMatrixCodeReader](#). The given ESerializer must have been created for reading.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Save(  
    Euresys.Open_eVision_2_10.ESerializer serializer  
    )
```

Parameters

serializer

The serializer.

EMatrixCodeReader.SearchParams

Parameter space that the algorithm uses to read a Data Matrix code from an ROI.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ESearchParamsType SearchParams
{
    get;
}
```

Remarks

It can be modified through automatic learning (using the [EMatrixCodeReader::Learn](#) or [EMatrixCodeReader::LearnMore](#) methods) or by using [EMatrixCodeReader::SearchParams](#) properties and methods.

EMatrixCodeReader.SetIso29158CalibrationParameters

Sets ISO/IEC 29158 calibration paramters

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetIso29158CalibrationParameters (
    float Rcal,
    float MLcal,
    float SRcal,
    float SRtarget
)
```

Parameters

Rcal

Reported reflectance value, from a calibration standard.

MLcal

Mean of the light from a histogram of the calibrated standard.

SRcal

System response parameters(such as exposure and/or again) used to create an image of the calibration standard.

SRtarget

System response parameters(such as exposure and/or again) used to create an image of the symbole under test.

EMatrixCodeReader.SetLearnMaskElement

Allows to choose which decoded parameters are learnt when the [EMatrixCodeReader::Learn](#) or [EMatrixCodeReader::LearnMore](#) methods are called.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetLearnMaskElement(
    Euresys.Open_eVision_2_10.ELearnParam index,
    bool value
)
```

Parameters

index

Parameter identifier, as defined in [ELearnParam](#).

value

TRUE to enable the parameter for learning.

Remarks

In order to enable a parameter for learning, you need to set corresponding item of LearnMask to TRUE. Default: all items are set to TRUE.

EMatrixCodeReader.TimeOut

Time-out for the [EMatrixCodeReader::Learn](#), [EMatrixCodeReader::LearnMore](#) and **Read** methods.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint TimeOut  
  
    { get; set; }
```

Remarks

If the processing time of one of these functions becomes longer than the set time-out, the processing is stopped and an exception is thrown. In that case, the error code of the exception is [TimeoutReached](#). The time-out is set in microseconds. This time-out is not a real time-out. The processing is stopped as soon as possible after the time-out has been reached. This means that the time elapsed effectively in the method can be greater than the time-out in itself.

4.113. EMatrixCodeReader Class

An [EMatrixCodeReader](#) instance can detect, decode and grade matrixcodes in an ROI, it returns a vector of [EMatrixCode](#) instances.

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

Properties

ComputeGrading

Allows to choose whether the grading properties of the [EMatrixCode](#) object will be computed by the [EMatrixCodeReader::Read](#) method. The default setting for this property is `false`.

Iso29158CalibrationParameters

ISO/IEC TR 29158 calibration parameters

MaxNumCodes

The maximum number of codes that the reader should try to find.

ReadMode

The [EReadMode](#) used for the [EMatrixCodeReader::Read](#) method. The default value for this property is [Speed](#).

ReadResults

Outputs the codes that were found by the [EMatrixCodeReader::Read](#) method

TimeOut

M The timeout for the [EMatrixCodeReader::Read](#) and [EMatrixCodeReader::Learn](#) method in microseconds.

e

thods

[EMatrixCodeReader](#)

Creates an [EMatrixCodeReader](#) object.

[Learn](#)

Learns the optimal parameter settings for detecting data matrix codes in the given ROI

[Load](#)

Load the configuration for this [EMatrixCodeReader](#) instance.

[operator=](#)

Assignment operator

[Read](#)

Tries to locate, decode and read data matrix codes in the given ROI

[ResetLearning](#)

Forgets the learned parameter settings and resets their default values

[Save](#)

Save the configuration for this [EMatrixCodeReader](#) instance.

StopProcess

EM Stops the [EMatrixCodeReader::Read](#) and/or [EMatrixCodeReader::Learn](#) process as soon as possible.
M

EMatrixCodeReader.ComputeGrading

Allows to choose whether the grading properties of the [EMatrixCode](#) object will be computed by the [EMatrixCodeReader::Read](#) method. The default setting for this property is `false`.

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]  
  
bool ComputeGrading  
  
{ get; set; }
```

EMatrixCodeReader.EMatrixCodeReader

Creates an [EMatrixCodeReader](#) object.

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]  
  
void EMatrixCodeReader(  
)  
  
void EMatrixCodeReader(  
    Euresys.Open_eVision_2_10.EasyMatrixCode2.EMatrixCodeReader other  
)
```

Parameters

other

Another [EMatrixCodeReader](#) object to be copied in the new [EMatrixCodeReader](#) object.

EMatrixCodeReader.Iso29158CalibrationParameters

ISO/IEC TR 29158 calibration parameters

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]
Euresys.Open_eVision_2_10.EMatrixCodeIso29158CalibrationParameters
Iso29158CalibrationParameters
    { get; set; }
```

EMatrixCodeReader.Learn

Learns the optimal parameter settings for detecting data matrix codes in the given ROI

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]
void Learn(
    Euresys.Open_eVision_2_10.EROIBW8 roi
)
```

Parameters

roi

The ROI in which the data matrix codes have to be found.

EMatrixCodeReader.Load

Load the configuration for this [EMatrixCodeReader](#) instance.

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]
void Load(
    string path
)
```

Parameters

path

The path from which to load the configuration.

EMatrixCodeReader.MaxNumCodes

The maximum number of codes that the reader should try to find.

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]
uint MaxNumCodes
{ get; set; }
```

Remarks

By default, this parameter is set to 1. If this property is set to 0, the reader will try to find as many codes as possible.

EMatrixCodeReader.operator=

Assignment operator

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]
Euresys.Open_eVision_2_10.EasyMatrixCode2.EMatrixCodeReader operator=(
    Euresys.Open_eVision_2_10.EasyMatrixCode2.EMatrixCodeReader other
)
```

Parameters

other

[EMatrixCodeReader](#) object to be copied.

EMatrixCodeReader.Read

Tries to locate, decode and read data matrix codes in the given ROI

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]
void Read(
    Euresys.Open_eVision_2_10.EROIBW8 roi
)
```

Parameters

roi

The ROI in which the data matrix codes have to be found.

EMatrixCodeReader.ReadMode

The [EReadMode](#) used for the [EMatrixCodeReader::Read](#) method. The default value for this property is [Speed](#).

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]
Euresys.Open_eVision_2_10.EasyMatrixCode2.EReadMode ReadMode
{ get; set; }
```

EMatrixCodeReader.ReadResults

Outputs the codes that were found by the [EMatrixCodeReader::Read](#) method

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]
Euresys.Open_eVision_2_10.EasyMatrixCode2.EMatrixCode[] ReadResults
{ get; }
```

EMatrixCodeReader.ResetLearning

Forgets the learned parameter settings and resets their default values

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]
void ResetLearning(
)
```

EMatrixCodeReader.Save

Save the configuration for this [EMatrixCodeReader](#) instance.

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]
void Save(
    string path
)
```

Parameters

path

The path to which to save the configuration.

EMatrixCodeReader.StopProcess

Stops the [EMatrixCodeReader::Read](#) and/or [EMatrixCodeReader::Learn](#) process as soon as possible.

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]
```

```
void StopProcess (
)
```

Remarks

When this method is called the process is stopped at the first checkpoint.

EMatrixCodeReader.TimeOut

The timeout for the [EMatrixCodeReader::Read](#) and [EMatrixCodeReader::Learn](#) method in microseconds.

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

```
[C#]
uint TimeOut
{ get; set; }
```

Remarks

If the processing time of one of these methods becomes longer than the set time-out period, the process is stopped. The [EMatrixCodeReader::Read](#) method will return all the codes it has decoded up to that point. The [EMatrixCodeReader::Learn](#) method will only learn from those codes it has found within the time-out period. Note that the time-out period is not exact: the process is stopped at the first checkpoint after the time-out period has elapsed.

4.114. EMeasurementUnit Class

The measurement units that are supported by Open eVision.

Remarks

Measurement units are used to represent physical units, such as "meter" or "inch", and ease conversions between different unit systems. They are used to build dimensional values. The following length measurement units are predefined: **um** (microns), **mm**, **cm**, **dm**, **m**, **Dm**, **Hm**, **Km**, **mil** (1/1000 inch), **inch**, **foot**, **yard**, **mile**.

Namespace: Euresys.Open_eVision_2_10

Properties

Magnitude

Relative magnitude of this unit, with respect to a standard unit (e.g. 1 mm = 0.001 m).

Name

M Pointer to a **NULL**-terminated string containing the unit abbreviation.

e

Methods

ConversionFactorTo

Returns the factor needed to convert from this measurement unit to a second one.

EMeasurementUnit

Constructs a measurement unit.

GetStockMeasurementUnit

E

M

MeasurementUnit.ConversionFactorTo

Returns the factor needed to convert from this measurement unit to a second one.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float ConversionFactorTo(
    Euresys.Open_eVision_2_10.EMeasurementUnit Unit
)
```

Parameters

Unit

Reference to the second measurement unit.

EMeasurementUnit.EMeasurementUnit

Constructs a measurement unit.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EMeasurementUnit(
    float magnitude,
    string name
)

void EMeasurementUnit(
    Euresys.Open_eVision_2_10.EMeasurementUnit pUnit
)

void EMeasurementUnit(
)
```

Parameters

magnitude

Relative magnitude of this unit with respect to a standard (e.g. 1 mm = 0.001 m).

name

Unit abbreviation (e.g. "**mm**").

pUnit

-

EMeasurementUnit.GetStockMeasurementUnit

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EMeasurementUnit GetStockMeasurementUnit(
    Euresys.Open_eVision_2_10.EStockMeasurementUnit unit
)
```

Parameters

unit

-

EMeasurementUnit.Magnitude

Relative magnitude of this unit, with respect to a standard unit (e.g. 1 mm = 0.001 m).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float Magnitude
{ get; set; }
```

EMeasurementUnit.Name

Pointer to a **NULL**-terminated string containing the unit abbreviation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
string Name  
    { get; set; }
```

4.115. EMemorySerializer Class

Handles and EMemorySerializer context

Base Class: [ESerializer](#)

Namespace: Euresys.Open_eVision_2_10

Properties

Buffer	Address of the internal buffer.
BufferSize	Size of the internal buffer
CurrentPosition	Current position in the buffer
Writing	M Checks if the serializer is in writing mode

Methods

Close	Closes the serializer
-----------------------	-----------------------

EMemorySerializer.Buffer

Address of the internal buffer.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
IntPtr Buffer  
    { get; }
```

EMemorySerializer.BufferSize

Size of the internal buffer

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint BufferSize  
    { get; }
```

EMemorySerializer.Close

Closes the serializer

Namespace: Euresys.Open_eVision_2_10


```
[C#]  
void Close(  
    )
```

EMemorySerializer.CurrentPosition

Current position in the buffer

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint CurrentPosition  
    { get; }
```

EMemorySerializer.Writing

Checks if the serializer is in writing mode

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
override bool Writing  
    { get; }
```

4.116. EMesh Class

Represents a 3D meshed object (https://en.wikipedia.org/wiki/Triangle_mesh).

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

PointCloud

Returns the point cloud of the [EMesh](#) object.

TriangleCount

Returns the number of triangles.
If the [EMesh](#) object has no triangle mesh data, the method returns 0.

TriangleIndexes

M Returns a pointer to the index array (an array of 'int') representing the list of triangles. Indexes are referring to the array of [E3DPoint](#).
e A triangle is defined by 3 indexes (T1a T1b T1c T2a T2b T2c T3a ...). The triangle index array size is a multiple of 3.

Methods

Index values must be in $[0, N[$ where N is the number of points in the point cloud.
If the [EMesh](#) object has no triangle mesh data, this method returns NULL.

EMesh

Creates an [EMesh](#) object.

Load

Loads the 3D Object. The given [ESerializer](#) must have been created for reading.

LoadSTL

Loads a triangle mesh from a STL file ([https://en.wikipedia.org/wiki/STL_\(file_format\)](https://en.wikipedia.org/wiki/STL_(file_format)))

operator=

Assignment operator.

Save

Saves the 3D Object. The given [ESerializer](#) must have been created for writing.

SaveSTL

[-] Saves the triangle mesh to an STL file ([https://en.wikipedia.org/wiki/STL_\(file_format\)](https://en.wikipedia.org/wiki/STL_(file_format))). Only ASCII format is supported.

M

esh.EMesh

Creates an [EMesh](#) object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void EMesh(
)
void EMesh(
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud point_cloud
)
void EMesh(
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud point_cloud,
    int[] triangle_indexes
)
```

```
void EMesh(  
    Euresys.Open_eVision_2_10.Easy3D.EMesh other  
)
```

Parameters

point_cloud

A point cloud

triangle_indexes

The triangle mesh is a list of indexes, referring to the array contained in "point_cloud".

A triangle is defined by 3 indexes (T1a T1b T1c T2a T2b T2c T3a ...).

Index values must be in [0, N[with N the number of points in the [EPointCloud](#).

The triangle index array size is a multiple of 3.

other

Another [EMesh](#).

EMesh.Load

Loads the 3D Object. The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void Load(  
    Euresys.Open_eVision_2_10.ESerializer serializer  
)
```

Parameters

serializer

The serializer.

EMesh.LoadSTL

Loads a triangle mesh from a STL file ([https://en.wikipedia.org/wiki/STL_\(file_format\)](https://en.wikipedia.org/wiki/STL_(file_format)))

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void LoadSTL(
    string path
)
```

Parameters

path

The path to the STL file.

EMesh.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EMesh operator=(
    Euresys.Open_eVision_2_10.Easy3D.EMesh other
)
```

Parameters

other

-

EMesh.PointCloud

Returns the point cloud of the [EMesh](#) object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EPointCloud PointCloud
{
    get;
}
```

EMesh.Save

Saves the 3D Object. The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The serializer.

EMesh.SaveSTL

Saves the triangle mesh to an STL file ([https://en.wikipedia.org/wiki/STL_\(file_format\)](https://en.wikipedia.org/wiki/STL_(file_format))). Only ASCII format is supported.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SaveSTL(
    string path
)
```

Parameters

path

The path to the STL file.

EMesh.TriangleCount

Returns the number of triangles.
If the [EMesh](#) object has no triangle mesh data, the method returns 0.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
int TriangleCount
{ get; }
```

EMesh.TriangleIndexes

Returns a pointer to the index array (an array of 'int') representing the list of triangles. Indexes are referring to the array of [E3DPoint](#).

A triangle is defined by 3 indexes (T1a T1b T1c T2a T2b T2c T3a). The triangle index array size is a multiple of 3. Index values must be in [0, N[where N is the number of points in the point cloud.

If the [EMesh](#) object has no triangle mesh data, this method returns NULL.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
IntPtr TriangleIndexes
    { get; }
```

4.117. EMeshToZMapConverter Class

Computes a [EZMap](#) from a [EPointCloud](#). The value of the pixels of the ZMap are the distance between the 3D points and the reference plane.

All 3D points under the reference plane are discarded.

Various options can be set with methods [EMeshToZMapConverter](#), [EMeshToZMapConverter::SetFillMode](#), [EMeshToZMapConverter](#), [EMeshToZMapConverter](#), [EMeshToZMapConverter...](#)

When the conversion is called without defining specific parameters, the algorithm uses the following options:

- The reference plane is the horizontal plane.
- The orientation vector is selected automatically.
- The origin is set as the lowest left position of the projected point cloud on the reference plane.
- The resolution (the dimensions of the Z map) is estimated to have approximately one Point Cloud point per ZMap pixels.
- The scale is calculated from the point cloud ranges and the estimated resolution.
- The fill mode is enabled and the method is set to 'EFillUndefinedPixelsDirection_Local' (see method [EDepthMap8::FillUndefinedPixels](#)).

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

Extension

Sets a metric value used to enlarge the point cloud 3D domain. That value affects X,Y and Z directions and can be used to generate an [EZMap](#) with borders of undefined pixels. Default value is **0**, which means a ZMap without border.

FillUndefinedPixelsDirection

Gets the undefined pixel fill direction (see [EFillUndefinedPixelsDirection](#)).

FillUndefinedPixelsMethod

Gets the undefined pixel fill method (see [EFillUndefinedPixelsMethod](#)).

MapHeight

Gets the required height (number of pixels) of the generated [EZMap](#).
By default, the required size is not set.

MapWidth

Gets the required width (number of pixels) of the generated [EZMap](#).
By default, the required size is not set.

MapXResolution

Gets the resolution of the [EZMap](#) pixels along the X axis.

MapYResolution

Gets the resolution of the [EZMap](#) pixels along the Y axis.

MapZResolution

Gets/sets the [EZMap](#) Z resolution, in world space units per gray value.
The resolution is used to compute the transformation of the world Z position to an integer 8, 16 or 32 bits pixel value.

OrientationVector

Sets an explicit orientation for the [EZMap](#).
Overrides the orientation mode given by the method [EMeshToZMapConverter](#).

OrientationVectorMode

Chooses the [EZMap](#) orientation from a list of pre-defined axis, automatic mode or user defined vector.
Use [EMeshToZMapConverter](#) to set an explicit orientation vector for the ZMap.

Origin

Chooses the [EZMap](#) origin. It is the 3D world position used as origin of the ZMap upper left pixel (0,0).

ReferencePlane

Sets the [E3DPlane](#) reference plane.
The resulting [EZMap](#) is the distance of the 3D points above that plane.
3D points below the reference plane are discarded.

ReferencePlaneMode

Sets an axis aligned reference plane.
Overrides the explicit reference plane given by the method [EMeshToZMapConverter](#).

WorldToZMapTransform

Explicitly sets the world to ZMap transformation.
"SetWorldToZMapTransform" overrides the settings done by [EMeshToZMapConverter](#), [EMeshToZMapConverter](#) and [EMeshToZMapConverter](#).
That [E3DTransformMatrix](#) transform expresses how the world positions are transformed to the [EZMap](#) space.
The matrix must be a rigid transformation (translation and rotation only).
The resolution and scales of the ZMap are defined by [EMeshToZMapConverter](#), [EMeshToZMapConverter](#) and [EMeshToZMapConverter](#) methods.

ZMapToWorldTransform

Methods

Explicitly sets the ZMap to World transformation. "SetZMapToWorldTransform" overrides the settings done by [EMeshToZMapConverter](#), [EMeshToZMapConverter](#) and [EMeshToZMapConverter](#). That [E3DTransformMatrix](#) transform expresses how the [EZMap](#) positions are transformed to the World space. The matrix must be a rigid transformation (translation and rotation only). The resolution and scales of the World are defined by [EMeshToZMapConverter](#), [EMeshToZMapConverter](#) and [EMeshToZMapConverter](#) methods.

Convert

Computes a EZMap from a world space EMesh. The value of the pixels of the ZMap are the distance between the 3D triangles and the reference plane. Various options can be set with methods [EMeshToZMapConverter](#), [EMeshToZMapConverter](#), [EMeshToZMapConverter::SetMapSize](#), ...

EMeshToZMapConverter

Creates a [EMeshToZMapConverter](#) object.

EnableFillMode

Enables or disables fill mode. Fill mode parameters are defined by method [EMeshToZMapConverter::SetFillMode](#). Fill mode is enabled by default. If fill mode is disabled, undefined pixels may remain in the [EZMap](#).

IsFillModeEnabled

Tells if the fill mode is enabled or not. Use [EMeshToZMapConverter::EnableFillMode](#) to toggle the fill mode and [EMeshToZMapConverter::SetFillMode](#) to set the filling parameters.

Load	Loads the converter configuration. The given ESerializer must have been created for reading.
operator=	Assignment operator
operator==	Comparison operator
Save	Saves the converter configuration. The given ESerializer must have been created for writing.
SetFillMode	Inpainting options used to fill the "holes" in the EZMap . A hole exists when no 3D point is projected at that pixel position in the ZMap.
SetMapSize	Sets the required size of the generated EZMap ; expressed in number of pixels for width and height dimensions. By default, the required size is not set.
SetMapXYResolution	Sets the resolution (possibly anisotropic) of the EZMap pixels along the X and Y axes, in world space units per pixel (e.g mm/pixel). The resolution is used to compute the ZMap size (width and height), depending on the projected point cloud on the reference plane.
UnsetMapSize	Unsets the resolution. Lets the conversion decides for the optimum resolution, depending on the projected point cloud and pixel scale.

[UnsetMapXYResolution](#)

Unsets the X and Y resolutions. Lets the conversion decide the optimal resolution, depending on the projected point cloud and [EZMap](#) size.

[UnsetMapZResolution](#)

Unsets the ZMap Z resolution. Lets the conversion decide the optimal Z resolution.

[UnsetOrigin](#)

Lets the conversion process decides for the [EZMap](#) origin position (based on projected point cloud on the reference plane). Use [EMeshToZMapConverter](#) to enable and choose the ZMap origin.

[UnsetWorldToZMapTransform](#)

└─ Disables the explicit world to ZMap transformation, set with [EMeshToZMapConverter](#).

M

eshToZMapConverter.Convert

Computes a EZMap from a world space EMesh. The value of the pixels of the ZMap are the distance between the 3D triangles and the reference plane. Various options can be set with methods [EMeshToZMapConverter](#), [EMeshToZMapConverter](#), [EMeshToZMapConverter::SetMapSize](#), ...

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Convert(
    Euresys.Open_eVision_2_10.Easy3D.EMesh obj,
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 zmap
)
```

```
void Convert(  
    Euresys.Open_eVision_2_10.Easy3D.EMesh obj,  
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 zmap  
)  
  
void Convert(  
    Euresys.Open_eVision_2_10.Easy3D.EMesh obj,  
    Euresys.Open_eVision_2_10.Easy3D.EZMap32f zmap  
)
```

Parameters

obj

The input 3D mesh.

zmap

The generated ZMap in 8, 16 or 32 bits format.

EMeshToZMapConverter.EMeshToZMapConverter

Creates a [EMeshToZMapConverter](#) object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
void EMeshToZMapConverter(  
)  
  
void EMeshToZMapConverter(  
    Euresys.Open_eVision_2_10.Easy3D.EMeshToZMapConverter other  
)
```

Parameters

other

Reference to the [EMeshToZMapConverter](#) object used for the initialization.

EMeshToZMapConverter.EnableFillMode

Enables or disables fill mode. Fill mode parameters are defined by method [EMeshToZMapConverter::SetFillMode](#). Fill mode is enabled by default. If fill mode is disable, undefined pixels may remain in the [EZMap](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void EnableFillMode(
    bool state
)
```

Parameters

state

Set to true to enable fill mode.

EMeshToZMapConverter.Extension

Sets a metric value used to enlarge the point cloud 3D domain. That value affects X,Y and Z directions and can be used to generate an [EZMap](#) with borders of undefined pixels. Default value is **0**, which means a ZMap without border.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
float Extension
{ get; set; }
```

EMeshToZMapConverter.FillUndefinedPixelsDirection

Gets the undefined pixel fill direction (see [EFillUndefinedPixelsDirection](#)).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EFillUndefinedPixelsDirection
FillUndefinedPixelsDirection
    { get; }
```

EMeshToZMapConverter.FillUndefinedPixelsMethod

Gets the undefined pixel fill method (see [EFillUndefinedPixelsMethod](#)).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EFillUndefinedPixelsMethod FillUndefinedPixelsMethod
    { get; }
```

EMeshToZMapConverter.IsFillModeEnabled

Tells if the fill mode is enabled or not.

Use [EMeshToZMapConverter::EnableFillMode](#) to toggle the fill mode and [EMeshToZMapConverter::SetFillMode](#) to set the filling parameters.

Namespace: Euresys.Open_eVision_2_10.Easy3D


```
[C#]
bool IsFillModeEnabled(
)
```

EMeshToZMapConverter.Load

Loads the converter configuration. The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Load(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

Pointer to the [ESerializer](#) created for reading.

EMeshToZMapConverter.MapHeight

Gets the required height (number of pixels) of the generated [EZMap](#).
By default, the required size is not set.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
int MapHeight
```

```
{ get; }
```

EMeshToZMapConverter.MapWidth

Gets the required width (number of pixels) of the generated [EZMap](#).
By default, the required size is not set.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
int MapWidth
```

```
{ get; }
```

EMeshToZMapConverter.MapXResolution

Gets the resolution of the [EZMap](#) pixels along the X axis.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
float MapXResolution
```

```
{ get; }
```

EMeshToZMapConverter.MapYResolution

Gets the resolution of the [EZMap](#) pixels along the Y axis.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float MapYResolution  
{ get; }
```

EMeshToZMapConverter.MapZResolution

Gets/sets the [EZMap](#) Z resolution, in world space units per gray value.
The resolution is used to compute the transformation of the world Z position to an integer 8, 16 or 32 bits pixel value.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float MapZResolution  
{ get; set; }
```

EMeshToZMapConverter.operator=

Assignment operator

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EMeshToZMapConverter operator=(
    Euresys.Open_eVision_2_10.Easy3D.EMeshToZMapConverter other
)
```

Parameters

other

Reference to the [EMeshToZMapConverter](#) object used for the assignment.

EMeshToZMapConverter.operator==

Comparison operator

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool operator==(
    Euresys.Open_eVision_2_10.Easy3D.EMeshToZMapConverter other
)
```

Parameters

other

Reference to the [EMeshToZMapConverter](#) object used for the comparison.

EMeshToZMapConverter.OrientationVector

Sets an explicit orientation for the [EZMap](#).
Overrides the orientation mode given by the method [EMeshToZMapConverter](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
Euresys.Open_eVision_2_10.Easy3D.E3DPoint OrientationVector  
  
{ get; set; }
```

Remarks

The direction should be an [E3DPoint](#) representing the expected direction of the X (width) axis of the ZMap. That direction will be used after projection on the reference plane normal. That direction must NOT be aligned with the reference plane normal.

EMeshToZMapConverter.OrientationVectorMode

Chooses the [EZMap](#) orientation from a list of pre-defined axis, automatic mode or user defined vector. Use [EMeshToZMapConverter](#) to set an explicit orientation vector for the ZMap.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
Euresys.Open_eVision_2_10.Easy3D.EZMapOrientationVectorMode OrientationVectorMode  
  
{ get; set; }
```

Remarks

Choose between Automatic mode (default), world space axis or explicit user defined vector (see [EZMapOrientationVectorMode](#)).

EMeshToZMapConverter.Origin

Chooses the [EZMap](#) origin. It is the 3D world position used as origin of the ZMap upper left pixel (0,0).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DPoint Origin
{ get; set; }
```

Remarks

That position will be projected on the reference plane.

To let the conversion chooses for the origin, call [EMeshToZMapConverter::UnsetOrigin](#).

EMeshToZMapConverter.ReferencePlane

Sets the [E3DPlane](#) reference plane.
The resulting [EZMap](#) is the distance of the 3D points above that plane.
3D points below the reference plane are discarded.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DPlane ReferencePlane
{ get; set; }
```

EMeshToZMapConverter.ReferencePlaneMode

Sets an axis aligned reference plane.
Overrides the explicit reference plane given by the method [EMeshToZMapConverter](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
Euresys.Open_eVision_2_10.Easy3D.EZMapReferencePlaneMode ReferencePlaneMode
```

```
{ get; set; }
```

Remarks

Choose between X, Y or Z reference plane (see [EZMapReferencePlaneMode](#)).
The plane offset is set automatically on the point cloud lowest 3D point.

EMeshToZMapConverter.Save

Saves the converter configuration. The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
void Save(  
    Euresys.Open_eVision_2_10.ESerializer serializer  
)
```

Parameters

serializer

Pointer to the [ESerializer](#) created for writing.

EMeshToZMapConverter.SetFillMode

Inpainting options used to fill the "holes" in the [EZMap](#). A hole exists when no 3D point is projected at that pixel position in the ZMap.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetFillMode(
    Euresys.Open_eVision_2_10.Easy3D.EFillUndefinedPixelsDirection direction,
    Euresys.Open_eVision_2_10.Easy3D.EFillUndefinedPixelsMethod method
)
```

Parameters

direction

Direction in which the undefined pixels are filled in a depthmap from [EFillUndefinedPixelsDirection](#)

method

Which values used to fill the undefined pixels in a depthmap from [EFillUndefinedPixelsMethod](#)

EMeshToZMapConverter.SetMapSize

Sets the required size of the generated [EZMap](#); expressed in number of pixels for width and height dimensions. By default, the required size is not set.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetMapSize(
    int width,
    int height
)
```

Parameters

width

The required width for the Generated ZMap.

height

The required height for the Generated ZMap.

EMeshToZMapConverter.SetMapXYResolution

Sets the resolution (possibly anisotropic) of the [EZMap](#) pixels along the X and Y axes, in world space units per pixel (e.g mm/pixel).
The resolution is used to compute the ZMap size (width and height), depending on the projected point cloud on the reference plane.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetMapXYResolution(
    float resolution
)
void SetMapXYResolution(
    float resolutionX,
    float resolutionY
)
```

Parameters

resolution

The resolution for the isotropic case.

resolutionX

The resolution for the X axis.

resolutionY

The resolution for the Y axis.

Remarks

The isotropic scale, for X and Y axis is in metric world units.

EMeshToZMapConverter.UnsetMapSize

Unsets the resolution. Lets the conversion decides for the optimum resolution, depending on the projected point cloud and pixel scale.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void UnsetMapSize(  
)
```

EMeshToZMapConverter.UnsetMapXYResolution

Unsets the X and Y resolutions. Lets the conversion decide the optimal resolution, depending on the projected point cloud and [EZMap](#) size.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void UnsetMapXYResolution(  
)
```

EMeshToZMapConverter.UnsetMapZResolution

Unsets the ZMap Z resolution. Lets the conversion decide the optimal Z resolution.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void UnsetMapZResolution(  
)
```

EMeshToZMapConverter.UnsetOrigin

Lets the conversion process decides for the [EZMap](#) origin position (based on projected point cloud on the reference plane).

Use [EMeshToZMapConverter](#) to enable and choose the ZMap origin.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void UnsetOrigin(  
)
```

EMeshToZMapConverter.UnsetWorldToZMapTransform

Disables the explicit world to ZMap transformation, set with [EMeshToZMapConverter](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void UnsetWorldToZMapTransform(  
)
```

EMeshToZMapConverter.WorldToZMapTransform

Explicitly sets the world to ZMap transformation.

"SetWorldToZMapTransform" overrides the settings done by [EMeshToZMapConverter](#), [EMeshToZMapConverter](#) and [EMeshToZMapConverter](#).

That [E3DTransformMatrix](#) transform expresses how the world positions are transformed to the [EZMap](#) space. The matrix must be a rigid transformation (translation and rotation only).

The resolution and scales of the ZMap are defined by [EMeshToZMapConverter](#), [EMeshToZMapConverter](#) and [EMeshToZMapConverter](#) methods.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix WorldToZMapTransform  
  
{ get; set; }
```

EMeshToZMapConverter.ZMapToWorldTransform

Explicitly sets the ZMap to World transformation.

"SetZMapToWorldTransform" overrides the settings done by [EMeshToZMapConverter](#), [EMeshToZMapConverter](#) and [EMeshToZMapConverter](#).

That [E3DTransformMatrix](#) transform expresses how the [EZMap](#) positions are transformed to the World space. The matrix must be a rigid transformation (translation and rotation only).

The resolution and scales of the World are defined by [EMeshToZMapConverter](#), [EMeshToZMapConverter](#) and [EMeshToZMapConverter](#) methods.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix ZMapToWorldTransform  
  
{ get; set; }
```

4.118. EMovingAverage Class

Temporal integration of a number of images to reduce noise.

Namespace: Euresys.Open_eVision_2_10

Properties

SrcImage

M Returns the image in which you must store the next image to be averaged.

e

Methods

Average

Performs averaging of the source image with the preceding ones, and computes the de-noised image.

EMovingAverage

Constructs an EMovingAverage object.

GetSize

Queries a moving average context for the current parameter settings.

Reset

Restarts the average process as if no image had ever been handed to the EMovingAverage object.

SetSize

Initializes a moving average context with appropriate parameters.

EMovingAverage.Average

Performs averaging of the source image with the preceding ones, and computes the de-noised image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Average (
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage
)
void Average (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage
)
```

Parameters

destinationImage

Pointer to the destination image.

sourceImage

Pointer to the source image.

Remarks

The overload with only the destinationImage may be used only when the buffer allocation scheme has been set to internal (see [EMovingAverage::SetSize](#) or [EMovingAverage::EMovingAverage](#))

EMovingAverage.EMovingAverage

Constructs an EMovingAverage object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EMovingAverage (
    Euresys.Open_eVision_2_10.EMovingAverage other
)

void EMovingAverage (
)

void EMovingAverage (
    uint period,
    int width,
    int height,
    bool internalAllocationScheme
)
```

Parameters

other

-

period

Number of images on which to integrate. A power of 2 is recommended.

width

Image width (all images used for averaging must be of the same size).

height

Image height (all images used for averaging must be of the same size).

internalAllocationScheme

Buffer allocation scheme. When **TRUE**, the moving average context provides the image to be acquired into (see member [EMovingAverage::SrcImage](#)).

Remarks

The default constructor constructs a void moving average context. A void moving average context has no internal buffers allocated and cannot be used for integration. Use the [EMovingAverage::SetSize](#) member after construction, or the initializing constructor instead. The sizing constructor constructs and initializes a moving average context.

EMovingAverage.GetSize

Queries a moving average context for the current parameter settings.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetSize(
    ref uint numberOfImages,
    ref int imageWidth,
    ref int imageHeight,
    ref bool internalAllocationScheme
)
```

Parameters

numberOfImages

Number of images on which to integrate.

imageWidth

Image width (all images used for averaging must be of the same size).

imageHeight

Image height (all images used for averaging must be of the same size).

internalAllocationScheme

Buffer allocation scheme. When **TRUE**, the moving average context provides the image to be acquired into (see member [EMovingAverage::SrcImage](#)).

EMovingAverage.Reset

Restarts the average process as if no image had ever been handed to the EMovingAverage object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Reset(
)
```

Remarks

The behavior is thus the same as after a [EMovingAverage::SetSize](#) operation.

EMovingAverage.SetSize

Initializes a moving average context with appropriate parameters.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetSize(
    uint numberOfImages,
    int imageWidth,
    int imageHeight,
    bool internalAllocationScheme
)
```

Parameters

numberOfImages

Number of images on which to integrate. A power of 2 is recommended.

imageWidth

Image width.

imageHeight

Image height.

internalAllocationScheme

Buffer allocation scheme. When **TRUE**, the moving average context provides the image to be acquired into (see member [EMovingAverage::SrcImage](#)).

EMovingAverage.SrcImage

Returns the image in which you must store the next image to be averaged.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EImageBW8 SrcImage  
  
{ get; }
```

Remarks

This method may be used only when the buffer allocation scheme has been set to internal (see [EMovingAverage::SetSize](#) or [EMovingAverage::EMovingAverage](#))

4.119. EObject Class

This class represents an object (blob) in an encoded image.

Remarks

This class inherits from the [ECodedElement](#) class and provides additional methods to access the holes of a particular object.

The extraction of the holes is lazy. This means that the holes are not computed before they get accessed. For this reason, the first access to the holes is slower than the subsequent accesses. On the other hand, the applications that do not make use of the holes are not penalized by the cost of hole extraction.

Base Class: [ECodedElement](#)

Namespace: [Euresys.Open_eVision_2_10](#)

Properties

HoleCount

M Returns the number of holes in the object.

e

Methods

GetHole

Returns a specified hole in the object.

EObject.GetHole

Returns a specified hole in the object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EHole GetHole(  
    uint index  
)
```

Parameters

index

The index of the hole of interest.

EObject.HoleCount

Returns the number of holes in the object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint HoleCount  
{ get; }
```

4.120. EObjectBasedCalibrationGenerator Class

Represents an object-based 3D calibration generator.

The class performs the computation of a calibration model based on the scan of the reference object.

Base Class: [ECalibrationGenerator](#)

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

[CalibrationObjectScaleX](#)

Returns the X axis scale of the calibration object.

[CalibrationObjectScaleY](#)

Returns the Y axis scale of the calibration object.

[CalibrationObjectScaleZ](#)

Returns the Z axis scale of the calibration object.

[CalibrationObjectSizeA](#)

Returns the 'A' size of the calibration object.

[CalibrationObjectSizeB](#)

Returns the 'B' size of the calibration object.

[CalibrationObjectSizeC](#)

Returns the 'C' size of the calibration object.

[NumCalibrationPasses](#)

Sets/Gets the number of calibration passes.

Each passes will refine the calibration model but the computation will be longer.

The number of passes is **1** by default.

PrecisionVsSpeedTradeOff

Sets/Gets the trade-off between precision and speed for the calibration.

The Precision vs speed trade-off mode from [EObjectBasedCalibrationPrecisionVsSpeedTradeOff](#) is **EObjectBasedCalibrationPrecisionVsSpeedTradeOff_Balanced** by default.

RangeX

Sets/Gets the 'X' axis range for the calibration object detection in the [EDepthMap](#).

If max value is smaller than min value, then max will not be used, we use depth map **width - 1** instead.

If min value is smaller than 0, then min will not be used, we use **0** instead.

RangeY

Sets/Gets the 'Y' axis range for the calibration object detection in the depth map.

If max value is smaller than min value, then max will not be used, we use depth map **height - 1** instead.

If min value is smaller than 0, then min will not be used, we use **0** instead.

RangeZ

M Sets/Gets the 'Z' axis range for the calibration object detection in the depth map.

e if max value is smaller than min value, then max will not be used, we use the maximum float value instead.

if min value is smaller than 0, then min will not be used, we use **0** instead.

thods

Compute

Computes an [EObjectBasedCalibrationModel](#) from the [EDepthMap](#) of the calibration object.

EObjectBasedCalibrationGenerator

Creates a [EObjectBasedCalibrationGenerator](#).

GetCalibrationObjectType

Gets the [EObjectBasedCalibrationType](#) used for the computation of the [EObjectBasedCalibrationModel](#).

The type of the object used for the calibration is **E3DObjectBasedCalibrationType_NotDefined** by default.

Load

Loads the parameters used by the object based calibration generator.

operator=

Assignment operator.

Save

Saves the parameters used by the object based calibration generator.

SetCalibrationObjectScale

Sets the scale of your target calibration model compared to a reference model.

Refer to the user guide for the [EObjectBasedCalibrationModel](#) reference design.

Use that value to set the unit of the calibrated positions in the point cloud.

The scale will affect the world coordinates after conversion of the [EDepthMap](#) to [EPointCloud](#).

For example, if "1 reference unit" is equal to "10 millimeters", set "10" as scale value.

Then applying the calibration will produce results in millimeters.

Changing these values affect the calibration object sizes defined by [EObjectBasedCalibrationGenerator::SetCalibrationObjectType](#).

SetCalibrationObjectType

Sets the [EObjectBasedCalibrationType](#) used for the computation of the [EObjectBasedCalibrationModel](#).

The type of the object used for the calibration is **E3DObjectBasedCalibrationType_NotDefined** by default.

EObjectBasedCalibrationGenerator.CalibrationObjectScaleX

Returns the X axis scale of the calibration object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float CalibrationObjectScaleX  
    { get; }
```

EObjectBasedCalibrationGenerator.CalibrationObjectScaleY

Returns the Y axis scale of the calibration object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float CalibrationObjectScaleY  
    { get; }
```

EObjectBasedCalibrationGenerator.CalibrationObjectScaleZ

Returns the Z axis scale of the calibration object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
float CalibrationObjectScaleZ  
  
{ get; }
```

EObjectBasedCalibrationGenerator.CalibrationObjectSizeA

Returns the 'A' size of the calibration object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
float CalibrationObjectSizeA  
  
{ get; }
```

EObjectBasedCalibrationGenerator.CalibrationObjectSizeB

Returns the 'B' size of the calibration object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
float CalibrationObjectSizeB  
  
{ get; }
```


EObjectBasedCalibrationGenerator.CalibrationObjectSizeC

Returns the 'C' size of the calibration object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
float CalibrationObjectSizeC
{ get; }
```

EObjectBasedCalibrationGenerator.Compute

Computes an [EObjectBasedCalibrationModel](#) from the [EDepthMap](#) of the calibration object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EObjectBasedCalibrationModel Compute (
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 dm
)
Euresys.Open_eVision_2_10.Easy3D.EObjectBasedCalibrationModel Compute (
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 dm
)
Euresys.Open_eVision_2_10.Easy3D.EObjectBasedCalibrationModel Compute (
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap32f dm
)
```

Parameters

dm

The input depth map of the calibration object.

EObjectBasedCalibrationGenerator.EObjectBasedCalibrationGenerator

Creates a [EObjectBasedCalibrationGenerator](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void EObjectBasedCalibrationGenerator (
)
void EObjectBasedCalibrationGenerator (
    Euresys.Open_eVision_2_10.Easy3D.EObjectBasedCalibrationGenerator other
)
```

Parameters

other

The other [EObjectBasedCalibrationGenerator](#).

EObjectBasedCalibrationGenerator.GetCalibrationObjectType

Gets the [EObjectBasedCalibrationType](#) used for the computation of the [EObjectBasedCalibrationModel](#).
The type of the object used for the calibration is **E3DObjectBasedCalibrationType_NotDefined** by default.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EObjectBasedCalibrationType GetCalibrationObjectType (
)
```

EObjectBasedCalibrationGenerator.Load

Loads the parameters used by the object based calibration generator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Load(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#).

EObjectBasedCalibrationGenerator.NumCalibrationPasses

Sets/Gets the number of calibration passes.

Each passes will refine the calibration model but the computation will be longer.

The number of passes is **1** by default.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
int NumCalibrationPasses
{ get; set; }
```

EObjectBasedCalibrationGenerator.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EObjectBasedCalibrationGenerator operator=(
    Euresys.Open_eVision_2_10.Easy3D.EObjectBasedCalibrationGenerator other
)
```

Parameters

other

The other [EObjectBasedCalibrationGenerator](#).

EObjectBasedCalibrationGenerator.PrecisionVsSpeedTradeOff

Sets/Gets the trade-off between precision and speed for the calibration.

The Precision vs speed trade-off mode from [EObjectBasedCalibrationPrecisionVsSpeedTradeOff](#) is **EObjectBasedCalibrationPrecisionVsSpeedTradeOff_Balanced** by default.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EObjectBasedCalibrationPrecisionVsSpeedTradeOff
PrecisionVsSpeedTradeOff
{ get; set; }
```

EObjectBasedCalibrationGenerator.RangeX

Sets/Gets the 'X' axis range for the calibration object detection in the [EDepthMap](#).
If max value is smaller than min value, then max will not be used, we use depth map **width - 1** instead.
If min value is smaller than 0, then min will not be used, we use **0** instead.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.EIntegerRange RangeX  
  
{ get; set; }
```

EObjectBasedCalibrationGenerator.RangeY

Sets/Gets the 'Y' axis range for the calibration object detection in the depth map.
If max value is smaller than min value, then max will not be used, we use depth map **height - 1** instead.
If min value is smaller than 0, then min will not be used, we use **0** instead.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.EIntegerRange RangeY  
  
{ get; set; }
```

EObjectBasedCalibrationGenerator.RangeZ

Sets/Gets the 'Z' axis range for the calibration object detection in the depth map.
if max value is smaller than min value, then max will not be used, we use the maximum float value instead.
if min value is smaller than 0, then min will not be used, we use **0** instead.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.EFloatRange RangeZ
    { get; set; }
```

EObjectBasedCalibrationGenerator.Save

Saves the parameters used by the object based calibration generator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Save (
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#).

EObjectBasedCalibrationGenerator.SetCalibrationObjectScale

Sets the scale of your target calibration model compared to a reference model. Refer to the user guide for the [EObjectBasedCalibrationModel](#) reference design. Use that value to set the unit of the calibrated positions in the point cloud. The scale will affect the world coordinates after conversion of the [EDepthMap](#) to [EPointCloud](#). For example, if "1 reference unit" is equal to "10 millimeters", set "10" as scale value. Then applying the calibration will produce results in millimeters. Changing these values affect the calibration object sizes defined by [EObjectBasedCalibrationGenerator::SetCalibrationObjectType](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetCalibrationObjectScale(
    float scale
)

void SetCalibrationObjectScale(
    float scaleX,
    float scaleY,
    float scaleZ
)
```

Parameters

scale

Sets the same scale on axis X, Y and Z relative to the calibration object reference size.

scaleX

Sets the scale on X axis relative to the calibration object reference size.

scaleY

Sets the scale on Y axis relative to the calibration object reference size.

scaleZ

Sets the scale on Z axis relative to the calibration object reference size.

EObjectBasedCalibrationGenerator.SetCalibrationObjectType

Sets the [EObjectBasedCalibrationType](#) used for the computation of the [EObjectBasedCalibrationModel](#). The type of the object used for the calibration is **E3DObjectBasedCalibrationType_NotDefined** by default.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetCalibrationObjectType (
    Euresys.Open_eVision_2_10.Easy3D.EObjectBasedCalibrationType type
)

void SetCalibrationObjectType (
    Euresys.Open_eVision_2_10.Easy3D.EObjectBasedCalibrationType type,
    float sizeA,
    float sizeB,
    float sizeC
)
```

Parameters

type

Sets the type of object calibration to detect in the [EDepthMap](#).

sizeA

Sets the size 'A' of object calibration (**1** by default).

sizeB

Sets the size 'B' of object calibration (**1** by default).

sizeC

Sets the size 'C' of object calibration (**1** by default).

Remarks

'sizeA', 'sizeB', and 'sizeC' values are in metric unit. The resulting point cloud positions after applying the calibration will follow the same metric unit. Refer to the Open eVision user guide, Easy3D calibration section, for the definition of 'A', 'B' and 'C' dimensions.

4.121. EObjectBasedCalibrationModel Class

[EObjectBasedCalibrationModel](#) is an Easy3D calibration model built from a scan of a reference object.

Base Class: [ECalibrationModel](#)

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

[CalibrationError](#)

Returns the estimate of the calibration error (in metric units).

[CalibrationRelativeError](#)

Returns the estimate of the calibration error (in relative units).

[Type](#)

M Returns the type of calibration model, see [ECalibrationType](#).

e

Methods

[EObjectBasedCalibrationModel](#)

Creates an [EObjectBasedCalibrationModel](#).

[IsInitialized](#)

Returns true if the model is initialized.

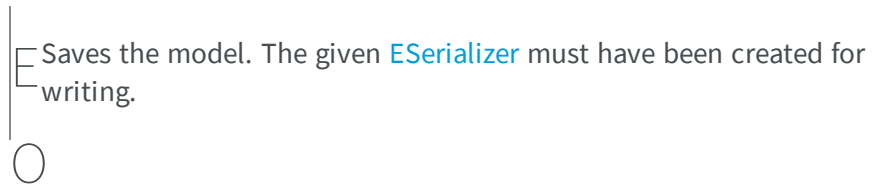
[Load](#)

Loads the model. The given [ESerializer](#) must have been created for reading.

[operator=](#)

Assignment operator.

Save



bjectBasedCalibrationModel.CalibrationError

Returns the estimate of the calibration error (in metric units).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float CalibrationError  
    { get; }
```

EObjectBasedCalibrationModel.CalibrationRelativeError

Returns the estimate of the calibration error (in relative units).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float CalibrationRelativeError  
    { get; }
```

EObjectBasedCalibrationModel.EObjectBasedCalibrationModel

Creates an [EObjectBasedCalibrationModel](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void EObjectBasedCalibrationModel (
)
void EObjectBasedCalibrationModel (
    Euresys.Open_eVision_2_10.Easy3D.EObjectBasedCalibrationModel other
)
```

Parameters

other

Another [EObjectBasedCalibrationModel](#).

EObjectBasedCalibrationModel.IsInitialized

Returns true if the model is initialized.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool IsInitialized (
)
```

EObjectBasedCalibrationModel.Load

Loads the model. The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Load(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The serializer.

EObjectBasedCalibrationModel.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EObjectBasedCalibrationModel operator=(
    Euresys.Open_eVision_2_10.Easy3D.EObjectBasedCalibrationModel other
)
```

Parameters

other

Another [EObjectBasedCalibrationModel](#).

EObjectBasedCalibrationModel.Save

Saves the model. The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The serializer.

EObjectBasedCalibrationModel.Type

Returns the type of calibration model, see [ECalibrationType](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override Euresys.Open_eVision_2_10.Easy3D.ECalibrationType Type
{ get; }
```

4.122. EObjectRunsIterator Class

Iterator to the runs of a coded element.

Remarks

This class is responsible for the sequential access to the individual runs of a coded element.

A run is defined as a maximal sequence of consecutive pixels on the same run, that all belong to the same coded element.

Namespace: Euresys.Open_eVision_2_10

Properties

EndX Returns the abscissa (X-coordinate) of the rightmost pixel of the run the iterator is currently over.

IsDone Tests whether all the runs have been spanned.

Length Returns the lengths of the run the iterator is currently over.

StartX Returns the abscissa (X-coordinate) of the leftmost pixel of the run the iterator is currently over.

Y Returns the ordinate (Y-coordinate) of the run the iterator is currently over.

Methods

EObjectRunsIterator Constructs an iterator to the runs of a coded element.

First Rewinds to the first run of the coded element.

Next

Advance to the next run in the iterator.

operator=

Assignment operator.

O

bjectRunsIterator.EndX

Returns the abscissa (X-coordinate) of the rightmost pixel of the run the iterator is currently over.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
int EndX  
  
{ get; }
```

Remarks

An exception is thrown if the iterator has reached its end. Use [EObjectRunsIterator::IsDone](#) to check this condition.

EObjectRunsIterator.EObjectRunsIterator

Constructs an iterator to the runs of a coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void EObjectRunsIterator(  
    )  
void EObjectRunsIterator(  
    Euresys.Open_eVision_2_10.ECodedElement codedElement  
    )  
void EObjectRunsIterator(  
    Euresys.Open_eVision_2_10.EObjectRunsIterator other  
    )
```

Parameters

codedElement

The coded element of interest, in the case the iterator is to be constructed from a given coded element.

other

The iterator to be copied, in the case of the copy constructor.

EObjectRunsIterator.First

Rewinds to the first run of the coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void First(  
    )
```

EObjectRunsIterator.IsDone

Tests whether all the runs have been spanned.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
bool IsDone
{ get; }
```

EObjectRunslterator.Length

Returns the lengths of the run the iterator is currently over.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint Length
{ get; }
```

Remarks

An exception is thrown if the iterator has reached its end. Use [EObjectRunslterator::IsDone](#) to check this condition.

EObjectRunslterator.Next

Advance to the next run in the iterator.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Next (
)
```

Remarks

An exception is thrown if the iterator has reached its end. Use [EObjectRunsIterator::IsDone](#) to check this condition.

EObjectRunsIterator.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EObjectRunsIterator operator=(
    Euresys.Open_eVision_2_10.EObjectRunsIterator other
)
```

Parameters

other

The iterator to be copied.

EObjectRunsIterator.StartX

Returns the abscissa (X-coordinate) of the leftmost pixel of the run the iterator is currently over.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int StartX
{ get; }
```

Remarks

An exception is thrown if the iterator has reached its end. Use [EObjectRunsIterator::IsDone](#) to check this condition.

EObjectRunIterator.Y

Returns the ordinate (Y-coordinate) of the run the iterator is currently over.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
int Y  
  
{ get; }
```

Remarks

An exception is thrown if the iterator has reached its end. Use [EObjectRunIterator::IsDone](#) to check this condition.

4.123. EObjectSelection Class

This container class handles the selection of a subset of coded elements taken from a coded image.

Remarks

This class provides methods to perform a selection of objects or holes and to retrieve the features of the coded elements in the collection.

Namespace: Euresys.Open_eVision_2_10

Properties

AttachedImage

Sets the attached image for computing the features that depend on an image.

ElementCount

Returns the number of coded elements selection.

FeretAngle

M Angle of the Feret box (in the current angle units).

e

thods

Add

Add a coded element to the selection.

AddHole

Adds a single hole of a coded image.

AddHoles

Adds all the holes contained in an object, in a layer or in a coded image.

AddHolesOfSelectedObjects

Adds the holes of all the objects that are currently selected.

AddLayer

Adds all the objects of a single layer in a coded image.

AddObject

Adds a single object of a layer in a coded image.

AddObjects

Adds all the objects of all the layers of a given coded image.

AddObjectsUsingFloatFeature

Selects the objects that fulfill a condition on a floating-point feature.

AddObjectsUsingIntegerFeature

Selects the objects that fulfill a condition on an integer feature.

AddObjectsUsingRectangle

Adds all the objects of a coded image whose location fulfill a criterion based on a rectangle.

AddObjectsUsingUnsignedIntegerFeature

Selects the objects that fulfill a condition on an unsigned integer feature.

AddObjectUsingPosition

Adds the object of a coded image that lies at a particular location.

Clear

Remove all the coded elements that are contained in the selection.

ClearFeatureCache

Clears the internal cache for the computed features.

EObjectSelection

-

FeatureAverage

Computes the average of the values of the given feature across the selection.

FeatureDeviation

Computes the standard deviation of the values of the given feature across the selection.

FeatureVariance

Computes the variance of the values of the given feature across the selection.

FloatFeatureMaximum

Computes the maximum value of the given feature across the selection.

FloatFeatureMinimum

Computes the minimum value of the given feature across the selection.

GetElement

Index-based access to the coded elements of the selection.

GetFloatFeature

Returns the value of a floating-point feature for a selected coded element.

GetIndexOfElement

Retrieves the index of a given coded element in the selection.

GetIntegerFeature

Returns the value of an integer feature for a selected coded element.

GetUnsignedIntegerFeature

Returns the value of an unsigned integer feature for a selected coded element.

IntegerFeatureMaximum

Computes the maximum value of the given feature across the selection.

IntegerFeatureMinimum

Computes the minimum value of the given feature across the selection.

IsSelected

Tests whether a given coded element is present in the selection.

Remove

Remove a coded element from the selection.

RemoveHole

Removes a single hole of a coded image.

RemoveHoles

Removes all the holes contained in an object, in a layer or in a coded image.

RemoveLayer

Removes all the objects of a single layer in a coded image.

RemoveObject

Removes a single object of a layer in a coded image.

RemoveObjectsUsingRectangle

Removes all the objects of a coded image whose location fulfill a criterion based on a rectangle.

RemoveObjectUsingPosition

Removes the object of a coded image that lies at a particular location.

RemoveSelectedHoles

Remove all the holes that are currently present in the selection.

RemoveUsingFloatFeature

Removes the selected coded elements that fulfill a condition on a floating-point feature.

RemoveUsingIntegerFeature

Removes the selected coded elements that fulfill a condition on an unsigned integer feature.

RemoveUsingUnsignedIntegerFeature

Removes the selected coded elements that fulfill a condition on an unsigned integer feature.

RenderMask

Creates a Flexible Mask from the selection.

Sort

Sorts the selected coded elements according to the value of a feature.

UnsignedIntegerFeatureMaximum

Computes the maximum value of the given feature across the selection.

UnsignedIntegerFeatureMinimum

EObjectSelection.Add

Computes the minimum value of the given feature across the selection.
Add a coded element to the selection.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Add(
    Euresys.Open_eVision_2_10.ECodedElement element
)
```

Parameters

element

The coded element.

EObjectSelection.AddHole

Adds a single hole of a coded image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddHole(
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    uint objectIndex,
    uint holeIndex
)

void AddHole(
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    uint layerIndex,
    uint objectIndex,
    uint holeIndex
)
```

Parameters

codedImage

The coded image.

objectIndex

The index of the parent object in the layer.

holeIndex

The index of the hole in the parent object.

layerIndex

The index of the layer of interest. If this parameter is left unspecified, the default layer will be taken into consideration.

Remarks

This methods throws an exception if no layer index is specified and if, simultaneously, the coded image contains several layers. Indeed, in such a case, no default layer exists.

EObjectSelection.AddHoles

Adds all the holes contained in an object, in a layer or in a coded image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddHoles (
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage
)
void AddHoles (
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    uint layerIndex
)
void AddHoles (
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    uint layerIndex,
    uint objectIndex
)
```

Parameters

codedImage

The source coded image.

layerIndex

The index of the layer of interest. If this parameter is left unspecified, all the layers are taken into consideration.

objectIndex

The index of the parent object. If this parameter is left unspecified, all the objects are taken into consideration.

EObjectSelection.AddHolesOfSelectedObjects

Adds the holes of all the objects that are currently selected.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddHolesOfSelectedObjects (
)
```

EObjectSelection.AddLayer

Adds all the objects of a single layer in a coded image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddLayer (
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    uint layerIndex
)
void AddLayer (
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage
)
```

Parameters

codedImage

The coded image.

layerIndex

The index of the layer of interest. If this parameter is left unspecified, the default layer will be taken into consideration.

Remarks

This methods throws an exception if no layer index is specified and if, simultaneously, the coded image contains several layers. Indeed, in such a case, no default layer exists.

EObjectSelection.AddObject

Adds a single object of a layer in a coded image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddObject(
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    uint objectIndex
)
void AddObject(
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    uint layerIndex,
    uint objectIndex
)
```

Parameters

codedImage

The coded image.

objectIndex

The index of the object in the layer.

layerIndex

The index of the layer of interest. If this parameter is left unspecified, the default layer will be taken into consideration.

Remarks

This methods throws an exception if no layer index is specified and if, simultaneously, the coded image contains several layers. Indeed, in such a case, no default layer exists.

EObjectSelection.AddObjects

Adds all the objects of all the layers of a given coded image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddObjects (
    Euresys.Open_eVision_2_10.ECodedImage2 image
)
```

Parameters

image

The coded image.

EObjectSelection.AddObjectsUsingFloatFeature

Selects the objects that fulfill a condition on a floating-point feature.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddObjectsUsingFloatFeature (
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    uint layerIndex,
    Euresys.Open_eVision_2_10.EFeature feature,
    float threshold,
    Euresys.Open_eVision_2_10.ESingleThresholdMode mode
)

void AddObjectsUsingFloatFeature (
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    uint layerIndex,
    Euresys.Open_eVision_2_10.EFeature feature,
    float lowBound,
    float highBound,
    Euresys.Open_eVision_2_10.EDoubleThresholdMode mode
)
```

```

void AddObjectsUsingFloatFeature(
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    Euresys.Open_eVision_2_10.EFeature feature,
    float threshold,
    Euresys.Open_eVision_2_10.ESingleThresholdMode mode
)

void AddObjectsUsingFloatFeature(
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    Euresys.Open_eVision_2_10.EFeature feature,
    float lowBound,
    float highBound,
    Euresys.Open_eVision_2_10.EDoubleThresholdMode mode
)

```

Parameters

codedImage

The source coded image.

layerIndex

The index of the layer of interest. If left unspecified, all the layers are taken into consideration.

feature

The feature that serves as a filter.

threshold

The single threshold on the feature.

mode

Specifies the way the threshold is interpreted.

lowBound

The low bound of the range on the feature.

highBound

The high bound of the range on the feature.

Remarks

Most features are floating-point. These methods thus tend to be the most widely used.

EObjectSelection.AddObjectsUsingIntegerFeature

Selects the objects that fulfill a condition on an integer feature.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void AddObjectsUsingIntegerFeature (
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    uint layerIndex,
    Euresys.Open_eVision_2_10.EFeature feature,
    int threshold,
    Euresys.Open_eVision_2_10.ESingleThresholdMode mode
)

void AddObjectsUsingIntegerFeature (
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    uint layerIndex,
    Euresys.Open_eVision_2_10.EFeature feature,
    int lowBound,
    int highBound,
    Euresys.Open_eVision_2_10.EDoubleThresholdMode mode
)

void AddObjectsUsingIntegerFeature (
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    Euresys.Open_eVision_2_10.EFeature feature,
    int threshold,
    Euresys.Open_eVision_2_10.ESingleThresholdMode mode
)

void AddObjectsUsingIntegerFeature (
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    Euresys.Open_eVision_2_10.EFeature feature,
    int lowBound,
    int highBound,
    Euresys.Open_eVision_2_10.EDoubleThresholdMode mode
)
```

Parameters

codedImage

The source coded image.

layerIndex

The index of the layer of interest. If left unspecified, all the layers are taken into consideration.

feature

The feature that serves as a filter.

threshold

The single threshold on the feature.

mode

Specifies the way the threshold is interpreted.

lowBound

The low bound of the range on the feature.

highBound

The high bound of the range on the feature.

EObjectSelection.AddObjectsUsingRectangle

Adds all the objects of a coded image whose location fulfill a criterion based on a rectangle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddObjectsUsingRectangle (
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    int x,
    int y,
    uint width,
    uint height,
    Euresys.Open_eVision_2_10.ERectangleMode mode
)

void AddObjectsUsingRectangle (
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    uint layerIndex,
    int x,
    int y,
    uint width,
    uint height,
    Euresys.Open_eVision_2_10.ERectangleMode mode
)
```

Parameters

codedImage

The source coded image.

x

The X-coordinate of the top-left corner of the selection rectangle.

y

The Y-coordinate of the top-left corner of the selection rectangle.

width

The width of the selection rectangle.

height

The height of the selection rectangle.

mode

The comparison mode with respect to the selection rectangle.

layerIndex

If specified, only the specified layer is taken into consideration.

EObjectSelection.AddObjectsUsingUnsignedIntegerFeature

Selects the objects that fulfill a condition on an unsigned integer feature.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void AddObjectsUsingUnsignedIntegerFeature (  
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,  
    uint layerIndex,  
    Euresys.Open_eVision_2_10.EFeature feature,  
    uint threshold,  
    Euresys.Open_eVision_2_10.ESingleThresholdMode mode  
)  
  
void AddObjectsUsingUnsignedIntegerFeature (  
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,  
    Euresys.Open_eVision_2_10.EFeature feature,  
    uint threshold,  
    Euresys.Open_eVision_2_10.ESingleThresholdMode mode  
)
```

```
void AddObjectsUsingUnsignedIntegerFeature (
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    uint layerIndex,
    Euresys.Open_eVision_2_10.EFeature feature,
    uint lowBound,
    uint highBound,
    Euresys.Open_eVision_2_10.EDoubleThresholdMode mode
)

void AddObjectsUsingUnsignedIntegerFeature (
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    Euresys.Open_eVision_2_10.EFeature feature,
    uint lowBound,
    uint highBound,
    Euresys.Open_eVision_2_10.EDoubleThresholdMode mode
)
```

Parameters

codedImage

The source coded image.

layerIndex

The index of the layer of interest. If left unspecified, all the layers are taken into consideration.

feature

The feature that serves as a filter.

threshold

The single threshold on the feature.

mode

Specifies the way the threshold is interpreted.

lowBound

The low bound of the range on the feature.

highBound

The high bound of the range on the feature.

EObjectSelection.AddObjectUsingPosition

Adds the object of a coded image that lies at a particular location.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddObjectUsingPosition(
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    int x,
    int y
)
```

Parameters

codedImage

The source coded image.

x

The X-coordinate of the object.

y

The Y-coordinate of the object.

Remarks

If no object lies at the specified coordinate, the selection is not changed.

EObjectSelection.AttachedImage

Sets the attached image for computing the features that depend on an image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EROIBW8 AttachedImage
{ get; set; }
```

Remarks

An image must be attached before dealing with the following features: [PixelMin](#), [PixelMax](#), [WeightedGravityCenterX](#), [WeightedGravityCenterY](#), [PixelGrayAverage](#), [PixelGrayVariance](#), [PixelGrayDeviation](#).

EObjectSelection.Clear

Remove all the coded elements that are contained in the selection.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Clear(  
)
```

EObjectSelection.ClearFeatureCache

Clears the internal cache for the computed features.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void ClearFeatureCache(  
)
```

Remarks

This is useful to reduce memory consumption.

EObjectSelection.ElementCount

Returns the number of coded elements selection.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint ElementCount
{ get; }
```

EObjectSelection.EObjectSelection

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EObjectSelection(
    Euresys.Open_eVision_2_10.EObjectSelection other
)
void EObjectSelection(
)
```

Parameters

other

-

EObjectSelection.FeatureAverage

Computes the average of the values of the given feature across the selection.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float FeatureAverage(
    Euresys.Open_eVision_2_10.EFeature feature
)
```

Parameters

feature

The feature of interest.

EObjectSelection.FeatureDeviation

Computes the standard deviation of the values of the given feature across the selection.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float FeatureDeviation(
    Euresys.Open_eVision_2_10.EFeature feature
)
```

Parameters

feature

The feature of interest.

EObjectSelection.FeatureVariance

Computes the variance of the values of the given feature across the selection.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float FeatureVariance(
    Euresys.Open_eVision_2_10.EFeature feature
)
```

Parameters

feature

The feature of interest.

EObjectSelection.FeretAngle

Angle of the Feret box (in the current angle units).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float FeretAngle
{ get; set; }
```

Remarks

A Feret angle must be set for the following features: [FeretBoxCenterX](#), [FeretBoxCenterY](#), [FeretBoxWidth](#), [FeretBoxHeight](#).

EObjectSelection.FloatFeatureMaximum

Computes the maximum value of the given feature across the selection.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float FloatFeatureMaximum(
    Euresys.Open_eVision_2_10.EFeature feature
)
```

Parameters

feature

The feature of interest.

Remarks

Most features are floating-point. This method thus tends to be the most widely used.

EObjectSelection.FloatFeatureMinimum

Computes the minimum value of the given feature across the selection.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float FloatFeatureMinimum(
    Euresys.Open_eVision_2_10.EFeature feature
)
```

Parameters

feature

The feature of interest.

Remarks

Most features are floating-point. This method thus tends to be the most widely used.

EObjectSelection.GetElement

Index-based access to the coded elements of the selection.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ECodedElement GetElement(
    uint index
)
```

Parameters

index

The index of the coded element of interest.

EObjectSelection.GetFloatFeature

Returns the value of a floating-point feature for a selected coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float GetFloatFeature(
    uint index,
    Euresys.Open_eVision_2_10.EFeature feature
)
```

Parameters

index

The index of the selected coded element.

feature

The feature of interest.

Remarks

Most features are floating-point. This method thus tends to be the most widely used.

EObjectSelection.GetIndexOfElement

Retrieves the index of a given coded element in the selection.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
uint GetIndexOfElement(  
    Euresys.Open_eVision_2_10.ECodedElement element  
)
```

Parameters

element

The coded element of interest.

Remarks

An exception is thrown if the coded element is not present in the selection.

EObjectSelection.GetIntegerFeature

Returns the value of an integer feature for a selected coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
int GetIntegerFeature(  
    uint index,  
    Euresys.Open_eVision_2_10.EFeature feature  
)
```

Parameters

index

The index of the selected coded element.

feature

The feature of interest.

EObjectSelection.GetUnsignedIntegerFeature

Returns the value of an unsigned integer feature for a selected coded element.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint GetUnsignedIntegerFeature(  
    uint index,  
    Euresys.Open_eVision_2_10.EFeature feature  
)
```

Parameters

index

The index of the selected coded element.

feature

The feature of interest.

EObjectSelection.IntegerFeatureMaximum

Computes the maximum value of the given feature across the selection.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int IntegerFeatureMaximum(
    Euresys.Open_eVision_2_10.EFeature feature
)
```

Parameters

feature

The feature of interest.

EObjectSelection.IntegerFeatureMinimum

Computes the minimum value of the given feature across the selection.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int IntegerFeatureMinimum(
    Euresys.Open_eVision_2_10.EFeature feature
)
```

Parameters

feature

The feature of interest.

EObjectSelection.IsSelected

Tests whether a given coded element is present in the selection.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool IsSelected(
    Euresys.Open_eVision_2_10.ECodedElement element
)

bool IsSelected(
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    uint objectIndex
)

bool IsSelected(
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    uint layerIndex,
    uint objectIndex
)

bool IsSelected(
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    uint layerIndex,
    uint objectIndex,
    uint holeIndex
)
```

Parameters

element

The coded element.

codedImage

The coded image.

objectIndex

The index of the object in the layer of interest.

layerIndex

The index of the layer of interest. If this parameter is left unspecified, the default layer will be taken into consideration.

holeIndex

If specified, the index of the hole in the object. If unspecified, one tests the presence of the object.

Remarks

This methods throws an exception if no layer index is specified and if, simultaneously, the coded image contains several layers. Indeed, in such a case, no default layer exists.

EObjectSelection.Remove

Remove a coded element from the selection.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Remove (
    Euresys.Open_eVision_2_10.ECodedElement element
)
```

Parameters

element

The coded element.

EObjectSelection.RemoveHole

Removes a single hole of a coded image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void RemoveHole(  
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,  
    uint objectIndex,  
    uint holeIndex  
)  
  
void RemoveHole(  
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,  
    uint layerIndex,  
    uint objectIndex,  
    uint holeIndex  
)
```

Parameters

codedImage

The coded image.

objectIndex

The index of the parent object in the layer.

holeIndex

The index of the hole in the parent object.

layerIndex

The index of the layer of interest. If this parameter is left unspecified, the default layer will be taken into consideration.

Remarks

This methods throws an exception if no layer index is specified and if, simultaneously, the coded image contains several layers. Indeed, in such a case, no default layer exists.

EObjectSelection.RemoveHoles

Removes all the holes contained in an object, in a layer or in a coded image.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void RemoveHoles (
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage
)

void RemoveHoles (
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    uint layerIndex
)

void RemoveHoles (
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    uint layerIndex,
    uint objectIndex
)
```

Parameters

codedImage

The source coded image.

layerIndex

The index of the layer of interest. If this parameter is left unspecified, all the layers are taken into consideration.

objectIndex

The index of the parent object. If this parameter is left unspecified, all the objects are taken into consideration.

EObjectSelection.RemoveLayer

Removes all the objects of a single layer in a coded image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void RemoveLayer (
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage
)
```



```
void RemoveLayer(  
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,  
    uint layerIndex  
)
```

Parameters

codedImage

The coded image.

layerIndex

The index of the layer of interest. If this parameter is left unspecified, the default layer will be taken into consideration.

Remarks

This methods throws an exception if no layer index is specified and if, simultaneously, the coded image contains several layers. Indeed, in such a case, no default layer exists.

EObjectSelection.RemoveObject

Removes a single object of a layer in a coded image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void RemoveObject(  
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,  
    uint objectIndex  
)  
  
void RemoveObject(  
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,  
    uint layerIndex,  
    uint objectIndex  
)
```

Parameters

codedImage

The coded image.

objectIndex

The index of the object in the layer.

layerIndex

The index of the layer of interest. If this parameter is left unspecified, the default layer will be taken into consideration.

Remarks

This methods throws an exception if no layer index is specified and if, simultaneously, the coded image contains several layers. Indeed, in such a case, no default layer exists.

EObjectSelection.RemoveObjectsUsingRectangle

Removes all the objects of a coded image whose location fullfil a criterion based on a rectangle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void RemoveObjectsUsingRectangle(
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    int x,
    int y,
    uint width,
    uint height,
    Euresys.Open_eVision_2_10.ERectangleMode mode
)

void RemoveObjectsUsingRectangle(
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    uint layerIndex,
    int x,
    int y,
    uint width,
    uint height,
    Euresys.Open_eVision_2_10.ERectangleMode mode
)
```

Parameters

codedImage

The source coded image.

x

The X-coordinate of the top-left corner of the selection rectangle.

y

The Y-coordinate of the top-left corner of the selection rectangle.

width

The width of the selection rectangle.

height

The height of the selection rectangle.

mode

The comparison mode with respect to the selection rectangle.

layerIndex

If specified, only the specified layer is taken into consideration.

EObjectSelection.RemoveObjectUsingPosition

Removes the object of a coded image that lies at a particular location.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void RemoveObjectUsingPosition(
    Euresys.Open_eVision_2_10.ECodedImage2 codedImage,
    int x,
    int y
)
```

Parameters

codedImage

The source coded image.

x

The X-coordinate of the object.

y

The Y-coordinate of the object.

Remarks

If no object lies at the specified coordinate, the selection is not changed.

EObjectSelection.RemoveSelectedHoles

Remove all the holes that are currently present in the selection.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void RemoveSelectedHoles (  
)
```

EObjectSelection.RemoveUsingFloatFeature

Removes the selected coded elements that fulfill a condition on a floating-point feature.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void RemoveUsingFloatFeature (  
    Euresys.Open_eVision_2_10.EFeature feature,  
    float threshold,  
    Euresys.Open_eVision_2_10.ESingleThresholdMode mode  
)  
  
void RemoveUsingFloatFeature (  
    Euresys.Open_eVision_2_10.EFeature feature,  
    float low,  
    float high,  
    Euresys.Open_eVision_2_10.EDoubleThresholdMode mode  
)
```

Parameters

feature

The feature that serves as a filter.

threshold

The single threshold on the feature.

mode

Specifies the way the threshold is interpreted.

low

The low bound of the range on the feature.

high

The high bound of the range on the feature.

Remarks

Most features are floating-point. These methods thus tend to be the most widely used.

EObjectSelection.RemoveUsingIntegerFeature

Removes the selected coded elements that fulfill a condition on an unsigned integer feature.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void RemoveUsingIntegerFeature (
    Euresys.Open_eVision_2_10.EFeature feature,
    int threshold,
    Euresys.Open_eVision_2_10.ESingleThresholdMode mode
)

void RemoveUsingIntegerFeature (
    Euresys.Open_eVision_2_10.EFeature feature,
    int low,
    int high,
    Euresys.Open_eVision_2_10.EDoubleThresholdMode mode
)
```

Parameters

feature

The feature that serves as a filter.

threshold

The single threshold on the feature.

mode

Specifies the way the threshold is interpreted.

low

The low bound of the range on the feature.

high

The high bound of the range on the feature.

EObjectSelection.RemoveUsingUnsignedIntegerFeature

Removes the selected coded elements that fulfill a condition on an unsigned integer feature.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void RemoveUsingUnsignedIntegerFeature (
    Euresys.Open_eVision_2_10.EFeature feature,
    uint threshold,
    Euresys.Open_eVision_2_10.ESingleThresholdMode mode
)

void RemoveUsingUnsignedIntegerFeature (
    Euresys.Open_eVision_2_10.EFeature feature,
    uint low,
    uint high,
    Euresys.Open_eVision_2_10.EDoubleThresholdMode mode
)
```

Parameters

feature

The feature that serves as a filter.

threshold

The single threshold on the feature.

mode

Specifies the way the threshold is interpreted.

low

The low bound of the range on the feature.

high

The high bound of the range on the feature.

EObjectSelection.RenderMask

Creates a Flexible Mask from the selection.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void RenderMask (
    Euresys.Open_eVision_2_10.EROIBW8 result,
    int offsetX,
    int offsetY
)

void RenderMask (
    Euresys.Open_eVision_2_10.EROIBW8 result
)
```

Parameters

result

The image in which the generated mask will be stored.

offsetX

The X-offset that must be applied to bring the zero X-coordinate in the coded image on the first column of the result image (defaults to zero).

offsetY

The Y-offset that must be applied to bring the zero Y-coordinate in the coded image on the first row of the result image (defaults to zero).

Remarks

The size of the result image will not be changed: It must be properly sized beforehand.

This method generates an exception if several layers are encoded, in which case no default layer exists.

EObjectSelection.Sort

Sorts the selected coded elements according to the value of a feature.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Sort(
    Euresys.Open_eVision_2_10.EFeature feature
)
void Sort(
    Euresys.Open_eVision_2_10.EFeature feature,
    Euresys.Open_eVision_2_10.ESortDirection direction
)
```

Parameters

feature

The feature to sort with.

direction

The sorting direction. By default, the features are sorted by increasing values.

EObjectSelection.UnsignedIntegerFeatureMaximum

Computes the maximum value of the given feature across the selection.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint UnsignedIntegerFeatureMaximum(
    Euresys.Open_eVision_2_10.EFeature feature
)
```


Parameters

feature

The feature of interest.

EObjectSelection.UnsignedIntegerFeatureMinimum

Computes the minimum value of the given feature across the selection.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint UnsignedIntegerFeatureMinimum(
    Euresys.Open_eVision_2_10.EFeature feature
)
```

Parameters

feature

The feature of interest.

4.124. EOCR Class

Manages a complete context for the font-dependent printed character reader implemented in EasyOCR.

Namespace: Euresys.Open_eVision_2_10

Properties

[CharSpacing](#)

Spacing that separates characters.

CompareAspectRatio	Flag indicating whether the character aspect ratio is used to compute the recognition score.
CutLargeChars	Flag indicating whether the large chars cutting mode is used.
MatchingMode	Matching mode to use to compare characters to the template.
MaxCharHeight	Maximum character height.
MaxCharWidth	Maximum character width.
MinCharHeight	Minimum character height.
MinCharWidth	Minimum character width.
NoiseArea	Noise area.
NumChars	Number of recognized characters.
NumPatterns	Number of patterns in the current font.
PatternHeight	Current pattern height, as set at the last EOCR::NewFont or EOCR::Load operation.

PatternWidth

Current pattern width, as set at the last [EOCR::NewFont](#) or [EOCR::Load](#) operation.

RelativeSpacing

Relative spacing value.

RelativeThreshold

Relative threshold used for image segmentation.

RemoveBorder

Flag indicating whether all blobs touching the ROI edges are automatically discarded.

RemoveNarrowOrFlat

Flag indicating whether the characters are discarded when either dimension falls below the minimum value

SegmentationMode

Segmentation mode.

ShiftingMode

Shifting mode to use to compare characters to the template.

ShiftXTolerance

Horizontal translation tolerance around the nominal position when the character positions are explicitly specified.

ShiftYtolerance

Vertical translation tolerance around the nominal position when the character positions are explicitly specified.

TextColor

Text color.

Threshold

Threshold mode used for image segmentation.

TrueThreshold

M Absolute threshold level when using a single threshold.

e

thods

AddChar

Add a character coordinates to the list.

AddPatternFromImage

Adds a new pattern to the font.

BuildObjects

Segments the source image, i.e. detects and labels the objects.

CharGetDstX

Returns the abscissa of the lower right corner of a recognized character.

CharGetDstY

Returns the ordinate of the lower right corner of a recognized character.

CharGetHeight

Returns the height of a recognized character.

CharGetOrgX

Returns the abscissa of the upper left corner of a recognized character.

CharGetOrgY

Returns the ordinate of the upper left corner of a recognized character.

CharGetTotalDstX

Returns the abscissa of the lower right corner of a recognized character.

CharGetTotalDstY

Returns the ordinate of the lower right corner of a recognized character.

CharGetTotalOrgX

Returns the abscissa of the upper left corner of a recognized character.

CharGetTotalOrgY

Returns the ordinate of the upper left corner of a recognized character.

CharGetWidth

Returns the width of a recognized character.

DrawChar

Draws the bounding box of a given character.

DrawChars

Draws the bounding boxes of all characters in the image.

DrawCharsWithCurrentPen

Draws the bounding boxes of all characters in the image.

DrawCharWithCurrentPen

Draws the bounding box of a given character.

EmptyChars

Empties the list of known characters.

EOCR

Constructs an OCR context.

FindAllChars

Locates the characters by filtering the objects according to their size, and grouping them if the segmentation mode is set to [RepasteObjects](#).

GetConfidenceRatio

Returns the degree of confidence in the recognized character.

GetFirstCharCode

Returns the code of the pattern that matches at best a recognized character.

GetFirstCharDistance

Computes the degree of similarity between the best matching pattern and a recognized character.

GetPatternBitmap

Returns a pointer to an image holding the pattern of the given index. This image should not be modified.

GetPatternClass

Returns the class of a given pattern in the current font.

GetPatternCode

Returns the character code of a given pattern in the current font.

GetSecondCharCode

Returns the code of the pattern that matches at second best a recognized character.

GetSecondCharDistance

Computes the degree of similarity between the second best matching pattern and a recognized character.

HitChar

Returns **TRUE** if the cursor is placed over the character specified by **charIndex**.

HitChars

Returns **TRUE** if the cursor is placed over a character and sets the **charIndex** accordingly.

LearnPattern

Adds a new pattern to the font.

LearnPatterns

Adds all the patterns from the source image to the font.

Load

Loads the **EOCR** object configuration. The given **ESerializer** must have been created for reading.

MatchChar

Reads a single character, i.e. finds the best match between the patterns in the font and the given character.

NewFont

Empties the contents of the font and sets the size of the pattern array to be used from then on.

[operator=](#)

Copies all the data from another EOCR object into the current EOCR object

[ReadText](#)

Reads one or more rows of characters, i.e. finds the best match between the patterns in the font and the segmented characters.

[ReadTextWide](#)

Reads one or more rows of characters, i.e. finds the best match between the patterns in the font and the segmented characters.

[Recognize](#)

Achieves all processing phases (blob analysis, character segmentation and pattern recognition) in a single operation.

[RecognizeWide](#)

Achieves all processing phases (blob analysis, character segmentation and pattern recognition) in a single operation.

[RemovePattern](#)

Removes a given pattern from the current font.

[Save](#)

Saves the [EOCR](#) object configuration. The given [ESerializer](#) must have been created for writing.

[SetPatternClass](#)

Sets the class of a given pattern in the current font.

[SetPatternCode](#)

Sets the character code of a given pattern in the current font.

EOCR.AddChar

Add a character coordinates to the list.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddChar(
    int originX,
    int originY,
    int endX,
    int endY
)
```

Parameters

originX

Abscissa of the upper left corner of the bounding box of the character.

originY

Ordinate of the upper left corner of the bounding box of the character.

endX

Abscissa of the bottom right corner of the bounding box of the character.

endY

Ordinate of the bottom right corner of the bounding box of the character.

Remarks

It is to use when you know the exact position of the characters to be recognized, to bypass the segmentation step, for efficiency or reliability purposes. To use this feature, simply specify the character positions by successive calls to **AddChar**. When this is done, the remainder of the OCR processing steps can take place. To empty the list of known characters, call [EOCR::EmptyChars](#).

EOCR.AddPatternFromImage

Adds a new pattern to the font.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddPatternFromImage (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    int originX,
    int originY,
    int width,
    int height,
    int code,
    uint classes
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

originX

Abscissa of the upper left corner of the bounding box of the pattern.

originY

Ordinate of the upper left corner of the bounding box of the pattern.

width

Width of the bounding box of the pattern.

height

Height of the bounding box of the pattern.

code

Character code of the pattern.

classes

Class of the pattern, as defined by [EOCRClass](#).

Remarks

The pattern is extracted from an image by specifying a bounding rectangle.

EOCR.BuildObjects

Segments the source image, i.e. detects and labels the objects.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void BuildObjects (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

Remarks

An object is a connected set of pixels above or below **Threshold** (according to **TextColor**).

EOCR.CharGetDstX

Returns the abscissa of the lower right corner of a recognized character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int CharGetDstX (
    int index
)
```

Parameters

index

Character number (in range **0..NumChars-1**).

EOCR.CharGetDstY

Returns the ordinate of the lower right corner of a recognized character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int CharGetDstY(
    int index
)
```

Parameters

index

Character number (in range **0..NumChars-1**).

EOCR.CharGetHeight

Returns the height of a recognized character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int CharGetHeight(
    int index
)
```

Parameters

index

Character number (in range **0..NumChars-1**).

EOCR.CharGetOrgX

Returns the abscissa of the upper left corner of a recognized character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int CharGetOrgX(
    int index
)
```

Parameters

index

Character number (in range **0..NumChars-1**).

EOCR.CharGetOrgY

Returns the ordinate of the upper left corner of a recognized character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int CharGetOrgY(
    int index
)
```

Parameters

index

Character number (in range **0..NumChars-1**).

EOCR.CharGetTotalDstX

Returns the abscissa of the lower right corner of a recognized character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int CharGetTotalDstX(
    int index
)
```

Parameters

index

Character number (in range **0..NumChars-1**).

Remarks

The coordinates are computed with respect to the parent image rather than the ROI.

EOCR.CharGetTotalDstY

Returns the ordinate of the lower right corner of a recognized character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int CharGetTotalDstY(
    int index
)
```

Parameters

index

Character number (in range **0..NumChars-1**).

Remarks

The coordinates are computed with respect to the parent image rather than the ROI.

EOCR.CharGetTotalOrgX

Returns the abscissa of the upper left corner of a recognized character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
int CharGetTotalOrgX(  
    int index  
)
```

Parameters

index

Character number (in range **0..NumChars-1**).

Remarks

The coordinates are computed with respect to the parent image rather than the ROI.

EOCR.CharGetTotalOrgY

Returns the ordinate of the upper left corner of a recognized character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
int CharGetTotalOrgY(  
    int index  
)
```

Parameters

index

Character number (in range **0..NumChars-1**).

Remarks

The coordinates are computed with respect to the parent image rather than the ROI.

EOCR.CharGetWidth

Returns the width of a recognized character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int CharGetWidth(
    int index
)
```

Parameters

index

Character number (in range **0..NumChars-1**).

EOCR.CharSpacing

Spacing that separates characters.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int CharSpacing
{ get; set; }
```

Remarks

When two objects are separated by a vertical gap larger or equal than the spacing, they are considered to belong to distinct characters. The segmentation parameters *must* be the same for both learning and recognition process.

EOCR.CompareAspectRatio

Flag indicating whether the character aspect ratio is used to compute the recognition score.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool CompareAspectRatio  
    { get; set; }
```

Remarks

When **TRUE**, the character aspect ratio is used to compute the recognition score.

EOCR.CutLargeChars

Flag indicating whether the large chars cutting mode is used.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool CutLargeChars  
    { get; set; }
```

Remarks

If **TRUE**, candidate characters larger than **MaxWidth** are split into as many parts as required. If **FALSE**, large characters are discarded. The segmentation parameters *must* be the same for both learning and recognition process.

EOCR.DrawChar

Draws the bounding box of a given character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void DrawChar(
    IntPtr graphicContext,
    uint charIndex,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void DrawChar(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    uint charIndex,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void DrawChar(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    uint charIndex,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

charIndex

Character index, in range **0..NumChars-1**.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

color

The color in which to draw the overlay.

Remarks

Drawing is done in the device context associated to the desired window. The current pen is used.

EOCR.DrawChars

Draws the bounding boxes of all characters in the image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawChars (
    IntPtr graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void DrawChars (
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

```
void DrawChars (  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

color

The color in which to draw the overlay.

Remarks

Drawing is done in the device context associated to the desired window. The current pen is used for all characters (to vary the colors, draw the objects separately using the [EOCR::DrawChar](#) method instead).

EOCR.DrawCharsWithCurrentPen

Draws the bounding boxes of all characters in the image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawCharsWithCurrentPen(
    IntPtr graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

Remarks

Drawing is done in the device context associated to the desired window. The current pen is used for all characters (to vary the colors, draw the objects separately using the [EOCR::DrawChar](#) method instead).

EOCR.DrawCharWithCurrentPen

Draws the bounding box of a given character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void DrawCharWithCurrentPen(  
    IntPtr graphicContext,  
    uint charIndex,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

charIndex

Character index, in range **0..NumChars-1**.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

Remarks

Drawing is done in the device context associated to the desired window. The current pen is used.

EOCR.EmptyChars

Empties the list of known characters.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EmptyChars (
)
```

Remarks

It is to use when you know the exact position of the characters to be recognized. See member [EOCR::AddChar](#).

EOCR.EOCR

Constructs an OCR context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EOCR (
)
void EOCR (
    Euresys.Open_eVision_2_10.EOCR other
)
```

Parameters

other

Another EOCR object to be copied in the new EOCR object.

Remarks

By default, the **Threshold** property is set to **128**.

EOCR.FindAllChars

Locates the characters by filtering the objects according to their size, and grouping them if the segmentation mode is set to [RepasteObjects](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void FindAllChars(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage
)
```

Parameters

sourceImage

Pointer to the source image/ROI. This parameter is not used and kept only for compatibility purposes.

Remarks

The characters are sorted from left to right. This operation must be performed after a call to [EOCR::BuildObjects](#) and before a call to [EOCR::ReadText](#).

EOCR.GetConfidenceRatio

Returns the degree of confidence in the recognized character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float GetConfidenceRatio(
    int index
)
```


Parameters

index

Character number (in range **0..NumChars-1**).

Remarks

A value of 100 % means there is no difference between the recognized character and the best matching pattern. A value of 0 % means there is no way to distinguish between the best and second best matching pattern. The computation of the confidence ratio is based on the first and second characters distance parameters (see [EOCR::GetFirstCharDistance](#) and [EOCR::GetSecondCharDistance](#)).

EOCR.GetFirstCharCode

Returns the code of the pattern that matches at best a recognized character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int GetFirstCharCode(
    int index
)
```

Parameters

index

Character number (in range **0..NumChars-1**).

EOCR.GetFirstCharDistance

Computes the degree of similarity between the best matching pattern and a recognized character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float GetFirstCharDistance(
    int index
)
```

Parameters

index

Character number (in range **0..NumChars-1**).

Remarks

Returns **0** for a perfect match and **1** for a total mismatch.

EOCR.GetPatternBitmap

Returns a pointer to an image holding the pattern of the given index. This image should not be modified.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EImageBW8 GetPatternBitmap(
    int index
)
```

Parameters

index

Pattern number (in range **0.. NumPatterns -1**).

EOCR.GetPatternClass

Returns the class of a given pattern in the current font.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint GetPatternClass (
    int index
)
```

Parameters

index

Pattern number (in range **0..NumPatterns-1**).

EOCR.GetPatternCode

Returns the character code of a given pattern in the current font.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int GetPatternCode (
    int index
)
```

Parameters

index

Pattern number (in range **0..NumPatterns-1**).

EOCR.GetSecondCharCode

Returns the code of the pattern that matches at second best a recognized character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int GetSecondCharCode (
    int index
)
```

Parameters

index

Character number (in range **0..NumChars-1**).

EOCR.GetSecondCharDistance

Computes the degree of similarity between the second best matching pattern and a recognized character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float GetSecondCharDistance (
    int index
)
```

Parameters

index

Character number (in range **0..NumChars-1**).

Remarks

Returns **0** for a perfect match and **1** for a total mismatch.

EOCR.HitChar

Returns **TRUE** if the cursor is placed over the character specified by **charIndex**.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool HitChar(
    int cursorX,
    int cursorY,
    uint charIndex,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

Parameters

cursorX

Current cursor coordinates.

cursorY

Current cursor coordinates.

charIndex

Index of the character to be hit.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0**, the default value, the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

Remarks

If zooming and/or panning were used when drawing the ROI, the same values must be used with [EOCR::HitChar](#).

EOCR.HitChars

Returns **TRUE** if the cursor is placed over a character and sets the **charIndex** accordingly.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool HitChars (
    int cursorX,
    int cursorY,
    out uint charIndex,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

Parameters

cursorX

Current cursor coordinates.

cursorY

Current cursor coordinates.

charIndex

Index of the character hit.

zoomX

Horizontal zooming factor. By default, **TRUE** scale is used.

zoomY

Vertical zooming factor. If set to **0**, the default value, the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

Remarks

If zooming and/or panning were used when drawing the ROI, the same values must be used with [EOCR::HitChars](#).

EOCR.LearnPattern

Adds a new pattern to the font.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void LearnPattern(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    uint charIndex,
    int code,
    uint classes,
    bool autoSegmentation
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

charIndex

Index of the character (ASCII or Unicode) to learn.

code

Character code of the pattern.

classes

Class of the pattern, as defined by [EOCRClass](#).

autoSegmentation

Boolean indicating whether the calculation of the true threshold has to be forced (default **TRUE**) or bypassed (**FALSE**).

Remarks

The pattern is selected by its index value, as assigned by the [EOCR::FindAllChars](#) process.

EOCR.LearnPatterns

Adds all the patterns from the source image to the font.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void LearnPatterns(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    string text,  
    uint singleClass,  
    bool autoSegmentation  
)  
  
void LearnPatterns(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    string text,  
    uint[] classes,  
    bool autoSegmentation  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

text

String containing character codes of the patterns.

singleClass

If specified, all the characters of the string are associated with the same class(es), that is specified by **singleClass**.

autoSegmentation

Boolean indicating whether the calculation of the true threshold has to be forced (default **TRUE**) or bypassed (**FALSE**).

classes

If specified, the i-th character of the string is associated with the class specified by the i-th element of the vector **classes**.

Remarks

Patterns are ordered by their index value, as assigned by the [EOCR::FindAllChars](#) process.

EOCR.Load

Loads the [EOCR](#) object configuration. The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
void Load(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The serializer.

EOCR.MatchChar

Reads a single character, i.e. finds the best match between the patterns in the font and the given character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void MatchChar(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    uint classes,
    int index,
    int shiftX,
    int shiftY
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

classes

Logical mask obtained by combining the values of [EOCRClass](#), to specify to what classes the character may belong.

index

Character number (in range **0..NumChars-1**).

shiftX

Horizontal translation applied to the character.

shiftY

Vertical translation applied to the character.

Remarks

A shift can be apply to the character. This operation can only be performed after a call to [EOCR::FindAllChars](#).

EOCR.MatchingMode

Matching mode to use to compare characters to the template.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EMatchingMode MatchingMode
{ get; set; }
```

Remarks

By default, the root-mean-square error method is used. These modes are meant to be used without thresholding, that is when one of the [DarkOnLight](#) and [LightOnDark](#) colors are used.

EOCR.MaxCharHeight

Maximum character height.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int MaxCharHeight
{ get; set; }
```

Remarks

A character larger than the maximum width or higher than the maximum height is discarded. The segmentation parameters *must* be the same for both learning and recognition process.

EOCR.MaxCharWidth

Maximum character width.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
int MaxCharWidth  
  
    { get; set; }
```

Remarks

A character larger than the maximum width or higher than the maximum height is discarded. The segmentation parameters *must* be the same for both learning and recognition process.

EOCR.MinCharHeight

Minimum character height.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
int MinCharHeight  
  
    { get; set; }
```

Remarks

A character both narrower than the minimum width and lower than the minimum height is discarded. The segmentation parameters *must* be the same for both learning and recognition process.

EOCR.MinCharWidth

Minimum character width.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int MinCharWidth
{ get; set; }
```

Remarks

A character both narrower than the minimum width and lower than the minimum height is discarded. The segmentation parameters *must* be the same for both learning and recognition process.

EOCR.NewFont

Empties the contents of the font and sets the size of the pattern array to be used from then on.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void NewFont(
    uint patternWidth,
    uint patternHeight
)
```

Parameters

patternWidth

Width of the normalized pattern representation, in pixels.

patternHeight

Height of the normalized pattern representation, in pixels.

Remarks

A larger pattern array improves the recognition sensitivity, at the expense of increased processing time.

EOCR.NoiseArea

Noise area.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int NoiseArea
{ get; set; }
```

Remarks

When a blob has an area smaller than this value, it is ignored. The segmentation parameters *must* be the same for both learning and recognition process.

EOCR.NumChars

Number of recognized characters.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int NumChars
{ get; }
```

EOCR.NumPatterns

Number of patterns in the current font.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int NumPatterns
    { get; }
```

EOCR.operator=

Copies all the data from another EOCR object into the current EOCR object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EOCR operator=(
    Euresys.Open_eVision_2_10.EOCR other
)
```

Parameters

other
EOCR object to be copied

EOCR.PatternHeight

Current pattern height, as set at the last [EOCR::NewFont](#) or [EOCR::Load](#) operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint PatternHeight  
    { get; }
```

EOCR.PatternWidth

Current pattern width, as set at the last [EOCR::NewFont](#) or [EOCR::Load](#) operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint PatternWidth  
    { get; }
```

EOCR.ReadText

Reads one or more rows of characters, i.e. finds the best match between the patterns in the font and the segmented characters.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
string ReadText(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    int maximumNumberOfCharacters,  
    uint classes,  
    bool autoSegmentation  
)  
  
string ReadText(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    int maximumNumberOfCharacters,  
    uint[] classes,  
    bool autoSegmentation  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

maximumNumberOfCharacters

Maximum number of characters to be read.

classes

Pointer to an array of logical masks obtained by combining the values of [EOCRClass](#), to specify to what classes the character may belong. Each mask value in the array applies to the corresponding character.

autoSegmentation

Boolean indicating whether the calculation of the true threshold has to be forced (default **TRUE**) or bypassed (**FALSE**).

Remarks

This operation can only be performed after a call to [EOCR::FindAllChars](#). In case the text contains character with a code > 255, [EOCR::ReadText](#) will throw an exception. In this case, you should use [EOCR::ReadTextWide](#).

EOCR.ReadTextWide

Reads one or more rows of characters, i.e. finds the best match between the patterns in the font and the segmented characters.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
string ReadTextWide(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    int maximumNumberOfCharacters,
    uint classes,
    bool autoSegmentation
)

string ReadTextWide(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    int maximumNumberOfCharacters,
    uint[] classes,
    bool autoSegmentation
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

maximumNumberOfCharacters

Maximum number of characters to be read.

classes

Pointer to an array of logical masks obtained by combining the values of [EOCRClass](#), to specify to what classes the character may belong. Each mask value in the array applies to the corresponding character.

autoSegmentation

Boolean indicating whether the calculation of the true threshold has to be forced (default **TRUE**) or bypassed (**FALSE**).

Remarks

This operation can only be performed after a call to [EOCR::FindAllChars](#). In case the text contains character with a code > 65535, ReadTextWide will throw an exception. In this case, you should read the text character by character by using [EOCR::GetFirstCharCode](#).

EOCR.Recognize

Achieves all processing phases (blob analysis, character segmentation and pattern recognition) in a single operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
string Recognize(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    int maximumNumberOfCharacters,
    uint classes
)

string Recognize(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    int maximumNumberOfCharacters,
    uint[] classes
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

maximumNumberOfCharacters

Maximum number of characters to be read.

classes

Pointer to an array of logical mask obtained by combining the values of [EOCRClass](#), to specify to what classes the character may belong. Each mask value in the array applies to the corresponding character.

Remarks

This method does the same as a sequence of [EOCR::BuildObjects](#)/[EOCR::FindAllChars](#)/[EOCR::ReadText](#),

EOCR.RecognizeWide

Achieves all processing phases (blob analysis, character segmentation and pattern recognition) in a single operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
string RecognizeWide(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    int maximumNumberOfCharacters,  
    uint classes  
)  
  
string RecognizeWide(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    int maximumNumberOfCharacters,  
    uint[] classes  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

maximumNumberOfCharacters

Maximum number of characters to be read.

classes

Pointer to an array of logical mask obtained by combining the values of [EOCRClass](#), to specify to what classes the character may belong. Each mask value in the array applies to the corresponding character.

Remarks

This method does the same as a sequence of [EOCR::BuildObjects/EOCR::FindAllChars/EOCR::ReadText](#),

EOCR.RelativeSpacing

Relative spacing value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
float RelativeSpacing  
  
    { get; set; }
```

Remarks

This value is the ratio of the width of the spaces between characters and the character width. For example, characters 25 pixels wide separated by 10 pixels gaps correspond to a relative spacing of $10/25 = 0.4$. The default value of this parameter is **0**, corresponding to no spacing. This parameter is relevant only when the **CutLargeChars** mode is enabled. The segmentation parameters *must* be the same for both learning and recognition process.

EOCR.RelativeThreshold

Relative threshold used for image segmentation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float RelativeThreshold
{ get; set; }
```

Remarks

The **RelativeThreshold** is the fraction of the image pixels that will be set below the threshold. Only used when the [EOCR::Threshold](#) property is set to EThresholdMode_Relative. The default value is 0.5.

EOCR.RemoveBorder

Flag indicating whether all blobs touching the ROI edges are automatically discarded.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool RemoveBorder
{ get; set; }
```

Remarks

If **TRUE**, all blobs touching the ROI edges are automatically discarded. By default, this feature is turned on. The segmentation parameters *must* be the same for both learning and recognition process.

EOCR.RemoveNarrowOrFlat

Flag indicating whether the characters are discarded when either dimension falls below the minimum value

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool RemoveNarrowOrFlat
{ get; set; }
```

Remarks

If **TRUE**, characters are discarded if either their width or their height is smaller than the minimum value. By default, this feature is disabled (only smaller characters in *both* height and width are discarded – the condition could be written **Narrow** **And** **Flat**). The segmentation parameters *must* be the same for both learning and recognition process.

EOCR.RemovePattern

Removes a given pattern from the current font.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void RemovePattern(
    uint index
)
```

Parameters

index

Index of the pattern to be removed from the current font.

EOCR.Save

Saves the [EOCR](#) object configuration. The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The serializer.

EOCR.SegmentationMode

Segmentation mode.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ESegmentationMode SegmentationMode
{ get; set; }
```

Remarks

The segmentation parameters *must* be the same for both learning and recognition process.

EOCR.SetPatternClass

Sets the class of a given pattern in the current font.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetPatternClass (
    int index,
    uint classIdx
)
```

Parameters

index

Pattern number (in range **0..NumPatterns-1**).

classIdx

The class.

EOCR.SetPatternCode

Sets the character code of a given pattern in the current font.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetPatternCode (
    int index,
    int code
)
```

Parameters

index

Pattern number (in range **0..NumPatterns-1**).

code

The character code.

EOCR.ShiftingMode

Shifting mode to use to compare characters to the template.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EShiftingMode ShiftingMode
{ get; set; }
```

EOCR.ShiftXTolerance

Horizontal translation tolerance around the nominal position when the character positions are explicitly specified.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint ShiftXTolerance
{ get; set; }
```

Remarks

By default, no shifting is allowed (**ShiftX = 0**). The tolerance can be used in two ways: either each character is moved individually until the best match is found, or the set of characters is matched as a whole, until the best average error is reached. See [EOCR::ShiftingMode](#).

EOCR.ShiftYTolerance

Vertical translation tolerance around the nominal position when the character positions are explicitly specified.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint ShiftYTolerance  
  
{ get; set; }
```

Remarks

By default, no shifting is allowed (**ShiftY = 0**). The tolerance can be used in two ways: either each character is moved individually until the best match is found, or the set of characters is matched as a whole, until the best average error is reached. See [EOCR::ShiftingMode](#).

EOCR.TextColor

Text color.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EOCRColor TextColor  
  
{ get; set; }
```

Remarks

The segmentation parameters *must* be the same for both learning and recognition process.

EOCR.Threshold

Threshold mode used for image segmentation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int Threshold  
    { get; set; }
```

Remarks

Threshold value as defined by the EThresholdMode enum. By default, the "minimum residue" mode is used to determine the threshold value. In case of an absolute threshold, the threshold value is given instead.

EOCR.TrueThreshold

Absolute threshold level when using a single threshold.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint TrueThreshold  
    { get; }
```

Remarks

This value is valid only when the [EOCR::BuildObjects](#) or [EOCR::ReadText](#) method has been called beforehand.

4.125. EOCR2 Class

Manages a complete context for the font-dependent printed character reader implemented in EasyOCR2.

Namespace: Euresys.Open_eVision_2_10

Properties

CharacterDatabase

Sets the [EOCR2CharacterDatabase](#) used for recognizing text.

CharsHeight

Sets the expected character height in pixels.

CharsMaxFragmentation

Sets the **CharsMaxFragmentation** parameter for the segmentation algorithm. This will determine the minimum size a blob should be in order to be considered a potential character. A high setting will allow only larger blobs, a low setting will also allow smaller blobs.

The minimum blob size to be considered a potential character is defined as:

`CharsMaxFragmentation * CharsHeight * CharsWidth`

CharsSpacingBias

Sets the **CharSpacingBias** parameter for the topology fitting algorithm, which optimises the spacing of the bounding boxes to optimally fit the detected blobs. This will determine whether the method is biased toward finding narrow spacing, wide spacing or is neutral.

CharsWidthBias

Sets the **CharsWidthBias** parameter for the topology fitting algorithm, which optimises the width of the bounding boxes to optimally fit the detected blobs. This will determine whether the method is biased toward finding narrow boxes, wide boxes or is neutral.

CharsWidthRange

Sets the range of expected character widths in pixels.

DetectionDelta

Sets the **DetectionDelta** parameter for the segmentation algorithm. This will determine the range of grayscale-values used to determine the stability of a blob. A low setting will make the algorithm more sensitive to noise, a high setting will make the algorithm insensitive to blobs with low contrast to the background.

DetectionMethod

Sets the **EOCR2DetectionMethod** parameter for the topology fitting algorithm, which will place textboxes on the segmentation results, matching the given topology.

MaxVariation

Sets the **maxVariation** parameter for the segmentation algorithm. This parameter determines how stable a blob in the image should be in order to be considered a potential character, a region with clearly defined edges is generally considered stable while a blurry region is not. A high setting allows more unstable blobs, a low setting allows only very stable blobs.

NumDetectionPasses

Sets the **NumDetectionPasses** parameter for the topology fitting algorithm, which will place textboxes on the segmentation results (blobs), matching the given topology. The first pass will consider all blobs, subsequent passes will only consider those blobs that are inside the textboxes from the previous pass, sometimes resulting in a more optimal fit.

ReadText

Outputs an [EOCR2Text](#) structure containing the detailed detection and recognition results.

RelativeSpacesWidthRange

Sets the range of expected spaces between words as a fraction of the character width.

SegmentationMethod

Sets the **EOCR2SegmentationMethod** parameter for the segmentation algorithm, which will detect blobs in the image.

TextAngleRange

Sets the **TextAngleRange** parameter for the topology fitting algorithm, which will attempt to find the angle of the text in the image with respect to the horizontal. This will determine the center of the range of angles that will be tested, defined as:

```
TextAngleRange.min() <= angle <= TextAngleRange.max()
```

TextPolarity

Sets the **TextPolarity** parameter for the segmentation algorithm. This will determine whether the algorithm searches for light blobs in a dark background or for dark blobs in a light background.

TimeOut

Time-out for the [EOCR2::Read](#), [EOCR2::Detect](#) and [EOCR2::Recognize](#) methods.

Topology

thods

M Sets the topology of the text that should be found in the image. A modified version of Regex expressions are used, where:

e .(dot) represents any character (not including a space).

L represents a letter.

Lu represents an uppercase letter.

Ll represents a lowercase letter.

N represents a digit.

P represents a punctuation character !"#\$%()*,-./:;<?[_{}~

S represents the symbols \$+<=>|~

\n represents a line break.

' ' (space) represents a space between two words.

Combinations can be made, for example: `[LN]` represents an alpha-numeric character.

To specify multiple characters, simply add `{n}` at the end for n characters. If the amount of characters is uncertain, specify `{n,m}` for a minimum of n characters and a maximum of m characters.

The topology `"[LuN]{3,5}PN{4} \n .{5} LL"` represents a text comprised of 2 lines:

The first line has 1 word composed of 3 to 5 uppercase alpha-numeric characters, followed by a punctuation character and 4 numbers.

The second line has 2 words. The first word comprises of 5 wild-card characters, the second word has 2 alphabetic characters (upper- or lowercase).

AddCharactersToDatabase

Reads reference characters from disk and adds them to the database used for text recognition. The characters can be read from a trueType (.ttf/.ttc) file or from an EasyOCR2 database file.

ClearCharacterDatabase

Clears the reference character database from this [EOCR2](#) instance.

ClearResult

Clear the current detected text if it exists.

Detect

Finds the text in an image as follows:

- (1) Detects potential characters in the image following the given text polarity.
 - (2) Fits bounding boxes to the detected characters, following the given topology and character width/height.
 - (3) Extracts the detected characters from the image.
- The detected characters are output as an [EOCR2Text](#) structure.

DrawDetection

Draws the bounding boxes found by the topology detection algorithm.

DrawDetectionWithCurrentPen

Draws the bounding boxes found by the topology detection algorithm with the current pen.

DrawRecognition

Draws the recognized text next to the character bounding box in the image.

DrawRecognitionWithCurrentPen

Draws the recognized text next to the character bounding box in the image with the current pen.

DrawSegmentation

Draws the blobs found by the segmentation algorithm.

DrawSegmentationWithCurrentPen

Draws the blobs found by the segmentation algorithm with the current pen.

EOCR2

Constructs an [EOCR2](#) context.

HitTestChar

Tests the cursor position for the presence of a character. If one is present under the cursor, it returns true and fills the [EOCR2Char](#) object passed as parameter.

HitTestLine

Tests the cursor position for the presence of a line. If one is present under the cursor, it returns true and fills the [EOCR2Line](#) object passed as parameter.

HitTestText

Tests the cursor position for the presence of a text. If one is present under the cursor, it returns true and fills the [EOCR2Text](#) object passed as parameter.

HitTestWord

Tests the cursor position for the presence of a word. If one is present under the cursor, it returns true and fills the [EOCR2Word](#) object passed as parameter.

Learn

Learns reference characters from a given [EOCR2Text/EOCR2Line/EOCR2Word/EOCR2Char](#) instance, containing user-specified character codes.

Load

Loads a model, containing parameter settings used for all operations in [EOCR2](#), from disk.

operator=

Assignment operator, copies another [EOCR2](#) instance to this one.

Read	Performs all steps required for reading text from an image: (1) Detects potential characters in the image following the given text polarity and character width/height. (2) Fits bounding boxes to the detected characters, following the given topology and character width/height. (3) Recognizes the detected characters using the given reference character database. The read text is output as a string.
Recognize	Recognizes the characters in a given EOCR2Text instance, based on a given reference font.
Save	Saves the model to disk, containing the current parameter setting used for all operations in EOCR2 .
SaveCharacterDatabase	Saves the current reference character database to disk.

O

CR2.AddCharactersToDatabase

Reads reference characters from disk and adds them to the database used for text recognition. The characters can be read from a trueType (.ttf/.ttc) file or from an EasyOCR2 database file.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void AddCharactersToDatabase (  
    string file  
)
```

```
void AddCharactersToDatabase(  
    string file,  
    Euresys.Open_eVision_2_10.EasyOCR2CharacterFilter filter  
)
```

Parameters

- file*
A string containing the path of the file.
- filter*
Optional parameter; an [EasyOCR2CharacterFilter](#) that tells the method which subset of the character-set should be loaded.

EOCR2.CharacterDatabase

Sets the [EOCR2CharacterDatabase](#) used for recognizing text.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EOCR2CharacterDatabase CharacterDatabase  
{ get; set; }
```

Remarks

Setting a new characterDatabase will overwrite the current one.

EOCR2.CharsHeight

Sets the expected character height in pixels.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int CharsHeight
    { get; set; }
```

EOCR2.CharsMaxFragmentation

Sets the **CharsMaxFragmentation** parameter for the segmentation algorithm. This will determine the minimum size a blob should be in order to be considered a potential character. A high setting will allow only larger blobs, a low setting will also allow smaller blobs.

The minimum blob size to be considered a potential character is defined as:

$$\text{CharsMaxFragmentation} * \text{CharsHeight} * \text{CharsWidth}$$

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float CharsMaxFragmentation
    { get; set; }
```

Remarks

This parameter should be set between 0.0 and 1.0, the default setting is 0.1.

EOCR2.CharsSpacingBias

Sets the **CharSpacingBias** parameter for the topology fitting algorithm, which optimises the spacing of the bounding boxes to optimally fit the detected blobs. This will determine whether the method is biased toward finding narrow spacing, wide spacing or is neutral.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EasyOCR2CharSpacingBias CharsSpacingBias  
  
{ get; set; }
```

Remarks

The default setting for this parameter is [Neutral](#)

EOCR2.CharsWidthBias

Sets the **CharsWidthBias** parameter for the topology fitting algorithm, which optimises the width of the bounding boxes to optimally fit the detected blobs. This will determine whether the method is biased toward finding narrow boxes, wide boxes or is neutral.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EasyOCR2CharWidthBias CharsWidthBias  
  
{ get; set; }
```

Remarks

The default setting for this parameter is [Neutral](#)

EOCR2.CharsWidthRange

Sets the range of expected character widths in pixels.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EIntegerRange CharWidthRange  
  
{ get; set; }
```

Remarks

The CharWidthRange is returned by reference, changing it will affect the internal state of the [EOCR2](#) object.

EOCR2.ClearCharacterDatabase

Clears the reference character database from this [EOCR2](#) instance.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void ClearCharacterDatabase(  
)
```

EOCR2.ClearResult

Clear the current detected text if it exists.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void ClearResult(  
)
```

EOCR2.Detect

Finds the text in an image as follows:

- (1) Detects potential characters in the image following the given text polarity.
- (2) Fits bounding boxes to the detected characters, following the given topology and character width/height.
- (3) Extracts the detected characters from the image.

The detected characters are output as an [EOCR2Text](#) structure.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EOCR2Text Detect(
    Euresys.Open_eVision_2_10.EROIBW8 srcRoi
)

Euresys.Open_eVision_2_10.EOCR2Text Detect(
    Euresys.Open_eVision_2_10.EROIBW8 srcRoi,
    Euresys.Open_eVision_2_10.ERegion region
)
```

Parameters

srcRoi

The source image/ROI.

region

The region of interest where the detection is performed

Remarks

The variables Topology, CharHeight, CharWidth should be set before performing this operation. If the srcRoi is smaller than 3X3, an exception will be thrown.

EOCR2.DetectionDelta

Sets the **DetectionDelta** parameter for the segmentation algorithm. This will determine the range of grayscale-values used to determine the stability of a blob. A low setting will make the algorithm more sensitive to noise, a high setting will make the algorithm insensitive to blobs with low contrast to the background.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int DetectionDelta  
    { get; set; }
```

Remarks

This parameter should be set between 0 and 127, the default setting is 12.

EOCR2.DetectionMethod

Sets the **EOCR2DetectionMethod** parameter for the topology fitting algorithm, which will place textboxes on the segmentation results, matching the given topology.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EOCR2DetectionMethod DetectionMethod  
    { get; set; }
```

Remarks

The default setting for this parameter is [FixedWidth](#).

EOCR2.DrawDetection

Draws the bounding boxes found by the topology detection algorithm.

Namespace: Euresys.Open_eVision_2_10

```

[C#]
void DrawDetection(
    IntPtr hdc,
    Euresys.Open_eVision_2_10.EasyOCR2DrawDetectionStyle style,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)

void DrawDetection(
    IntPtr hdc,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.EasyOCR2DrawDetectionStyle style,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)

void DrawDetection(
    Euresys.Open_eVision_2_10.EDrawAdapter drawAdapter,
    Euresys.Open_eVision_2_10.EasyOCR2DrawDetectionStyle style,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)

```

Parameters

hdc

Handle of the device context on which to draw.

style

The style in which each detection box should be drawn.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

color

The color in which to draw the detection.

drawAdapter

-

EOCR2.DrawDetectionWithCurrentPen

Draws the bounding boxes found by the topology detection algorithm with the current pen.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawDetectionWithCurrentPen (
    IntPtr hdc,
    Euresys.Open_eVision_2_10.EasyOCR2DrawDetectionStyle style,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)
```

Parameters

hdc

Handle of the device context on which to draw.

style

The style in which each detection box should be drawn.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

EOCR2.DrawRecognition

Draws the recognized text next to the character bounding box in the image.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void DrawRecognition(  
    IntPtr hdc,  
    Euresys.Open_eVision_2_10.EasyOCR2DrawRecognitionStyle style,  
    uint cHeight,  
    float zoomX,  
    float zoomY,  
    int panX,  
    int panY  
)
```

```
void DrawRecognition(  
    Euresys.Open_eVision_2_10.EDrawAdapter drawAdapter,  
    Euresys.Open_eVision_2_10.EasyOCR2DrawRecognitionStyle style,  
    uint cHeight,  
    float zoomX,  
    float zoomY,  
    int panX,  
    int panY  
)
```

```
void DrawRecognition(  
    IntPtr hdc,  
    Euresys.Open_eVision_2_10.ERGBColor textColor,  
    Euresys.Open_eVision_2_10.ERGBColor backgroundColor,  
    Euresys.Open_eVision_2_10.EasyOCR2DrawRecognitionStyle style,  
    uint cHeight,  
    float zoomX,  
    float zoomY,  
    int panX,  
    int panY  
)
```

```
void DrawRecognition(  
    Euresys.Open_eVision_2_10.EDrawAdapter drawAdapter,  
    Euresys.Open_eVision_2_10.ERGBColor textColor,  
    Euresys.Open_eVision_2_10.ERGBColor backgroundColor,  
    Euresys.Open_eVision_2_10.EasyOCR2DrawRecognitionStyle style,  
    uint cHeight,  
    float zoomX,  
    float zoomY,  
    int panX,  
    int panY  
)
```

Parameters

hdc

Handle of the device context on which to draw.

style

The style in which each recognition result should be drawn.

cHeight

The character-height with which the recognized text should be displayed.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

drawAdapter

-

textColor

The color in which to draw the recognized text.

backgroundColor

The color of the box in which the text is displayed.

EOCR2.DrawRecognitionWithCurrentPen

Draws the recognized text next to the character bounding box in the image with the current pen.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawRecognitionWithCurrentPen (
    IntPtr hdc,
    Euresys.Open_eVision_2_10.EasyOCR2DrawRecognitionStyle style,
    uint cHeight,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)
```

Parameters

hdc

Handle of the device context on which to draw.

style

The style in which each recognition result should be drawn.

cHeight

The character-height with which the recognized text should be displayed.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

EOCR2.DrawSegmentation

Draws the blobs found by the segmentation algorithm.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void DrawSegmentation(
    IntPtr hdc,
    Euresys.Open_eVision_2_10.EasyOCR2DrawSegmentationStyle style,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)

void DrawSegmentation(
    IntPtr hdc,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.EasyOCR2DrawSegmentationStyle style,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)

void DrawSegmentation(
    Euresys.Open_eVision_2_10.EDrawAdapter drawAdapter,
    Euresys.Open_eVision_2_10.EasyOCR2DrawSegmentationStyle style,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)
```

Parameters

hdc

Handle of the device context on which to draw.

style

The style in which each blob should be drawn.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

color

The color in which to draw the segmentation.

drawAdapter

-

EOCR2.DrawSegmentationWithCurrentPen

Draws the blobs found by the segmentation algorithm with the current pen.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawSegmentationWithCurrentPen (
    IntPtr hdc,
    Euresys.Open_eVision_2_10.EasyOCR2DrawSegmentationStyle style,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)
```

Parameters

hdc

Handle of the device context on which to draw.

style

The style in which each blob should be drawn.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

EOCR2.EOCR2

Constructs an [EOCR2](#) context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EOCR2 (
)
void EOCR2 (
    Euresys.Open_eVision_2_10.EOCR2 other
)
```

Parameters

other

-

EOCR2.HitTestChar

Tests the cursor position for the presence of a character. If one is present under the cursor, it returns true and fills the [EOCR2Char](#) object passed as parameter.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool HitTestChar(
    Euresys.Open_eVision_2_10.EOCR2Char character,
    ref int lineN,
    ref int wordN,
    ref int charN,
    int x,
    int y,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)
```

Parameters

character

Returns the character if one was detected under the cursor.

lineN

Returns the line-number of the character if one was detected under the cursor.

wordN

Returns the word-number of the character if one was detected under the cursor.

charN

Returns the character-number of the character if one was detected under the cursor.

x

Horizontal position of the cursor.

y

Vertical position of the cursor.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

EOCR2.HitTestLine

Tests the cursor position for the presence of a line. If one is present under the cursor, it returns true and fills the [EOCR2Line](#) object passed as parameter.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool HitTestLine(
    Euresys.Open_eVision_2_10.EOCR2Line line,
    int x,
    int y,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)
```

Parameters

- line*
Object to fill if a line was detected under the cursor.
- x*
Horizontal position of the cursor.
- y*
Vertical position of the cursor.
- zoomX*
Horizontal zooming factor. By default, true scale is used.
- zoomY*
Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.
- panX*
Horizontal panning factor. By default, no panning occurs.
- panY*
Vertical panning factor. By default, no panning occurs.

EOCR2.HitTestText

Tests the cursor position for the presence of a text. If one is present under the cursor, it returns true and fills the [EOCR2Text](#) object passed as parameter.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool HitTestText(
    Euresys.Open_eVision_2_10.EOCR2Text text,
    int x,
    int y,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)
```

Parameters

- text*
Object to fill if a text was detected under the cursor.
- x*
Horizontal position of the cursor.
- y*
Vertical position of the cursor.
- zoomX*
Horizontal zooming factor. By default, true scale is used.
- zoomY*
Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.
- panX*
Horizontal panning factor. By default, no panning occurs.
- panY*
Vertical panning factor. By default, no panning occurs.

EOCR2.HitTestWord

Tests the cursor position for the presence of a word. If one is present under the cursor, it returns true and fills the [EOCR2Word](#) object passed as parameter.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool HitTestWord(
    Euresys.Open_eVision_2_10.EOCR2Word word,
    int x,
    int y,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)
```

Parameters

- word*
Object to fill if a word was detected under the cursor.
- x*
Horizontal position of the cursor.
- y*
Vertical position of the cursor.
- zoomX*
Horizontal zooming factor. By default, true scale is used.
- zoomY*
Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.
- panX*
Horizontal panning factor. By default, no panning occurs.
- panY*
Vertical panning factor. By default, no panning occurs.

EOCR2.Learn

Learns reference characters from a given [EOCR2Text](#)/[EOCR2Line](#)/[EOCR2Word](#)/[EOCR2Char](#) instance, containing user-specified character codes.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Learn(
    Euresys.Open_eVision_2_10.EOCR2Char character
)

void Learn(
    Euresys.Open_eVision_2_10.EOCR2Word word
)

void Learn(
    Euresys.Open_eVision_2_10.EOCR2Line line
)

void Learn(
    Euresys.Open_eVision_2_10.EOCR2Text text
)
```

Parameters

character

A single [EOCR2Char](#) character, containing the detected character from the reference image as well as the corresponding character code.

word

A single [EOCR2Word](#) word, containing the detected characters in a single word from the reference image as well as the corresponding character codes.

line

A single [EOCR2Line](#) line, containing the detected characters in a single line from the reference image as well as the corresponding character codes.

text

A complete [EOCR2Text](#) text, containing all detected characters from the reference image as well as the corresponding character codes.

Remarks

The [EOCR2Text](#)/[EOCR2Line](#)/[EOCR2Word](#)/[EOCR2Char](#) instance should contain detected characters from a reference image as well as their corresponding character codes.

EOCR2.Load

Loads a model, containing parameter settings used for all operations in [EOCR2](#), from disk.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void Load(  
    string modelPath  
)
```

Parameters

modelPath

A string containing the full path to the model file.

EOCR2.MaxVariation

Sets the **maxVariation** parameter for the segmentation algorithm. This parameter determines how stable a blob in the image should be in order to be considered a potential character, a region with clearly defined edges is generally considered stable while a blurry region is not. A high setting allows more unstable blobs, a low setting allows only very stable blobs.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
float MaxVariation
```

```
{ get; set; }
```

Remarks

This parameter should be set between 0.0 and 1.0, the default setting is 0.25.

EOCR2.NumDetectionPasses

Sets the **NumDetectionPasses** parameter for the topology fitting algorithm, which will place textboxes on the segmentation results (blobs), matching the given topology. The first pass will consider all blobs, subsequent passes will only consider those blobs that are inside the textboxes from the previous pass, sometimes resulting in a more optimal fit.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
int NumDetectionPasses  
  
{ get; set; }
```

Remarks

The default setting for this parameter is 1, the setting can be either 1 or 2.

EOCR2.operator=

Assignment operator, copies another [EOCR2](#) instance to this one.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EOCR2 operator=(  
    Euresys.Open_eVision_2_10.EOCR2 other  
)
```

Parameters

other

The [EOCR2](#) instance to copy from.

EOCR2.Read

Performs all steps required for reading text from an image:

- (1) Detects potential characters in the image following the given text polarity and character width/height.
- (2) Fits bounding boxes to the detected characters, following the given topology and character width/height.
- (3) Recognizes the detected characters using the given reference character database.

The read text is output as a string.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
string Read(  
    Euresys.Open_eVision_2_10.EROIBW8 srcRoi  
)  
  
string Read(  
    Euresys.Open_eVision_2_10.EROIBW8 srcRoi,  
    Euresys.Open_eVision_2_10.ERegion region  
)
```

Parameters

srcRoi

The source image/ROI.

region

The region of interest where the reading is performed

Remarks

The variables `TextPolarity`, `Topology`, `CharHeight`, `CharWidth` should be set and a reference character database should be set before performing this operation. If the `srcRoi` is smaller than 3X3, an exception will be thrown.

EOCR2.ReadText

Outputs an [EOCR2Text](#) structure containing the detailed detection and recognition results.

Namespace: `Euresys.Open_eVision_2_10`

```
[C#]
Euresys.Open_eVision_2_10.EOCR2Text ReadText
{ get; }
```

EOCR2.Recognize

Recognizes the characters in a given [EOCR2Text](#) instance, based on a given reference font.

Namespace: `Euresys.Open_eVision_2_10`

```
[C#]
string Recognize(
    Euresys.Open_eVision_2_10.EOCR2Text text
)
string Recognize(
    Euresys.Open_eVision_2_10.EOCR2Text text,
    Euresys.Open_eVision_2_10.EROIBW8 srcRoi
)
```

Parameters

text

[EOCR2Text](#) structure containing the detected characters from the image.

srcRoi

The source image/ROI.

Remarks

A reference character database should be provided before performing this operation. The overloaded function that uses an ROI is not yet implemented.

EOCR2.RelativeSpacesWidthRange

Sets the range of expected spaces between words as a fraction of the character width.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EFloatRange RelativeSpacesWidthRange  
{ get; set; }
```

Remarks

This parameter only affects the detection when the detectionMethod is set to [FixedWidth](#). The [RelativeSpacesWidthRange](#) is returned by reference, changing it will affect the internal state of the [EOCR2](#) object.

EOCR2.Save

Saves the model to disk, containing the current parameter setting used for all operations in [EOCR2](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void Save(  
    string modelPath  
)
```

Parameters

modelPath

A string containing the full path to the model file.

Remarks

It is advised to use a file extension that is non-standard (for instance *.ocr2)

EOCR2.SaveCharacterDatabase

Saves the current reference character database to disk.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void SaveCharacterDatabase(  
    string file  
)
```

Parameters

file

A string containing the path of the file.

EOCR2.SegmentationMethod

Sets the **EOCR2SegmentationMethod** parameter for the segmentation algorithm, which will detect blobs in the image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EOCR2SegmentationMethod SegmentationMethod
{ get; set; }
```

Remarks

The default setting for this parameter is [Local](#).

EOCR2.TextAngleRange

Sets the **TextAngleRange** parameter for the topology fitting algorithm, which will attempt to find the angle of the text in the image with respect to the horizontal. This will determine the center of the range of angles that will be tested, defined as:

```
TextAngleRange.min() <= angle <= TextAngleRange.max()
```

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EFloatRange TextAngleRange
{ get; set; }
```

Remarks

The `TextAngleRange` is returned by reference, changing it will affect the internal state of the [EOCR2](#) object. The default setting for this parameter is -20° to $+20^\circ$.

EOCR2.TextPolarity

Sets the **TextPolarity** parameter for the segmentation algorithm. This will determine whether the algorithm searches for light blobs in a dark background or for dark blobs in a light background.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EasyOCR2TextPolarity TextPolarity  
  
{ get; set; }
```

Remarks

Default setting is [WhiteOnBlack](#).

EOCR2.TimeOut

Time-out for the [EOCR2::Read](#), [EOCR2::Detect](#) and [EOCR2::Recognize](#) methods.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
System.UInt64 TimeOut  
  
{ get; set; }
```

Remarks

If the processing time of one of these functions becomes longer than the set time-out, the processing is stopped and an exception is thrown. In that case, the error code of the exception is [TimeoutReached](#). The time-out is set in microseconds. This time-out is not a real time-out. The processing is stopped as soon as possible after the time-out has been reached. This means that the time elapsed effectively in the method can be greater than the time-out in itself.

EOCR2.Topology

Sets the topology of the text that should be found in the image. A modified version of Regex expressions are used, where:

.(dot) represents any character (not including a space).

L represents a letter.

Lu represents an uppercase letter.

Ll represents a lowercase letter.

N represents a digit.

P represents a punctuation character !"#\$%'()*,-./:;<=>?[_{}~

S represents the symbols \$+<=>|~

\n represents a line break.

' ' (space) represents a space between two words.

Combinations can be made, for example: `[LN]` represents an alpha-numeric character.

To specify multiple characters, simply add `{n}` at the end for n characters. If the amount of characters is uncertain, specify `{n,m}` for a minimum of n characters and a maximum of m characters.

The topology `"[LuN]{3,5}PN{4} \n .{5} LL"` represents a text comprised of 2 lines:

The first line has 1 word composed of 3 to 5 uppercase alpha-numeric characters, followed by a punctuation character and 4 numbers.

The second line has 2 words. The first word comprises of 5 wildcard characters, the second word has 2 alphabetic characters (upper- or lowercase).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
string Topology
{ get; set; }
```

4.126. EOCR2Char Class

Holds all information related to a single detected character.

Namespace: Euresys.Open_eVision_2_10

Properties

Bitmap

The bitmap associated to this character.

BoundingBox

The bounding box associated to this character.

Candidates

The list of recognition results for all reference-characters with their respective scores.

Text

The true value of the character, this is used during the learning phase.

TextCode

M The true value of the character, this is used during the learning phase.

e

Methods

EOCR2Char

Constructs an [EOCR2Char](#) context.

operator=

E Copies all the data from another [EOCR2Char](#) object into the current [EOCR2Char](#) object

O

CR2Char.Bitmap

The bitmap associated to this character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EROIBW8 Bitmap  
  
{ get; }
```

EOCR2Char.BoundingBox

The bounding box associated to this character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.ERectangle BoundingBox  
  
{ get; }
```

EOCR2Char.Candidates

The list of recognition results for all reference-characters with their respective scores.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EOCR2CharacterCandidate[] Candidates  
  
{ get; }
```

EOCR2Char.EOCR2Char

Constructs an [EOCR2Char](#) context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EOCR2Char (
)
void EOCR2Char (
    Euresys.Open_eVision_2_10.EOCR2Char other
)
```

Parameters

other

EOCR2Char object to be copied.

EOCR2Char.operator=

Copies all the data from another [EOCR2Char](#) object into the current [EOCR2Char](#) object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EOCR2Char operator=(
    Euresys.Open_eVision_2_10.EOCR2Char other
)
```

Parameters

other

[EOCR2Char](#) object to be copied

EOCR2Char.Text

The true value of the character, this is used during the learning phase.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
string Text  
  
{ get; set; }
```

Remarks

It is also possible to set the character value using a UINT16 code, this is done with [EOCR2Char::TextCode](#).

EOCR2Char.TextCode

The true value of the character, this is used during the learning phase.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
ushort TextCode  
  
{ get; set; }
```

Remarks

It is also possible to set the character value using a std::string, this is done with [EOCR2Char::Text](#).

4.127. EOOCR2CharacterCluster Class

Holds all information related to character cluster.

Namespace: Euresys.Open_eVision_2_10

Properties

CharacterCount

Returns the number of characters in this cluster.

Characters

Returns all characters from this cluster.

Code

M The ASCII code of the cluster.

e

Methods

AddCharacter

Adds a character to the cluster.

Clear

Clears the cluster.

EOOCR2CharacterCluster

Constructs an [EOOCR2CharacterCluster](#) context.

GetCharacter

Returns a single character from the cluster.

operator=

Copies all the data from another [EOCR2CharacterCluster](#) object into the current [EOCR2CharacterCluster](#) object

[RemoveCharacter](#)

Removes a character from the cluster.

O

EOCR2CharacterCluster.AddCharacter

Adds a character to the cluster.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddCharacter (
    Euresys.Open_eVision_2_10.EOCR2DatabaseCharacter character
)
```

Parameters

character

The [EOCR2DatabaseCharacter](#) to be added to the database.

EOCR2CharacterCluster.CharacterCount

Returns the number of characters in this cluster.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int CharacterCount
    { get; }
```

EOCR2CharacterCluster.Characters

Returns all characters from this cluster.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EOCR2DatabaseCharacter[] Characters
    { get; }
```

EOCR2CharacterCluster.Clear

Clears the cluster.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Clear(
    )
```

EOCR2CharacterCluster.Code

The ASCII code of the cluster.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
ref ushort Code
{ get; set; }
```

EOCR2CharacterCluster.EOCR2CharacterCluster

Constructs an [EOCR2CharacterCluster](#) context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EOCR2CharacterCluster (
)
void EOCR2CharacterCluster (
    Euresys.Open_eVision_2_10.EOCR2CharacterCluster other
)
```

Parameters

other

[EOCR2CharacterCluster](#) object to be copied.

EOCR2CharacterCluster.GetCharacter

Returns a single character from the cluster.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EOCR2DatabaseCharacter GetCharacter(
    int index
)
```

Parameters

index

The index of this character.

EOCR2CharacterCluster.operator=

Copies all the data from another [EOCR2CharacterCluster](#) object into the current [EOCR2CharacterCluster](#) object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EOCR2CharacterCluster operator=(
    Euresys.Open_eVision_2_10.EOCR2CharacterCluster other
)
```

Parameters

other

[EOCR2CharacterCluster](#) object to be copied

EOCR2CharacterCluster.RemoveCharacter

Removes a character from the cluster.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void RemoveCharacter (
    int index
)
```

Parameters

index

The index of the character to be removed.

4.128. EOCR2CharacterDatabase Class

Holds all information related to a character database.

Namespace: Euresys.Open_eVision_2_10

Properties

Characters

M Returns all character from the database.

e

Methods

AddCharacter

Adds a single character to the database.

AddCharacters

Add characters to the database.

AddCluster

Adds the characters from an [EOCR2CharacterCluster](#) to the database

AddClusters

Adds the characters from a vector of [EOCR2CharacterCluster](#) objects to the database

ClearDatabase

Clears the database.

ClusterDatabase

Performs a clustering on the database.

EOCR2CharacterDatabase

Constructs an [EOCR2CharacterDatabase](#) context.

GetCharacter

Returns a character from the database.

operator=

Copies all the data from another [EOCR2CharacterDatabase](#) object into the current [EOCR2CharacterDatabase](#) object

RemoveCharacter

Removes a character from the database.

Save

Saves the character database to disk.

EOCR2CharacterDatabase.AddCharacter

Adds a single character to the database.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddCharacter(
    Euresys.Open_eVision_2_10.EOCR2DatabaseCharacter character
)
void AddCharacter(
    Euresys.Open_eVision_2_10.EOCR2Char character
)
```

Parameters

character

The [EOCR2DatabaseCharacter](#) or [EOCR2Char](#) to be added to the database.

EOCR2CharacterDatabase.AddCharacters

Add characters to the database.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddCharacters(
    Euresys.Open_eVision_2_10.EOCR2DatabaseCharacter[] characters
)
void AddCharacters(
    string path
)
```

```
void AddCharacters(  
    string path,  
    Euresys.Open_eVision_2_10.EasyOCR2CharacterFilter filter  
)  
  
void AddCharacters(  
    Euresys.Open_eVision_2_10.EOCR2Word word  
)  
  
void AddCharacters(  
    Euresys.Open_eVision_2_10.EOCR2Line line  
)  
  
void AddCharacters(  
    Euresys.Open_eVision_2_10.EOCR2Text text  
)
```

Parameters

characters

A vector of [EOCR2DatabaseCharacter](#) objects to be added to this database.

path

The path on disk of the character database to be added to this database.

filter

An [EasyOCR2CharacterFilter](#) that tells the method which subset of the character-set should be loaded.

word

An [EOCR2Word](#) object to be added to this database.

line

An [EOCR2Line](#) object to be added to this database.

text

An [EOCR2Text](#) object to be added to this database.

EOCR2CharacterDatabase.AddCluster

Adds the characters from an [EOCR2CharacterCluster](#) to the database

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddCluster (
    Euresys.Open_eVision_2_10.EOCR2CharacterCluster cluster
)
```

Parameters

cluster

The [EOCR2CharacterCluster](#) to be added to the database.

EOCR2CharacterDatabase.AddClusters

Adds the characters from a vector of [EOCR2CharacterCluster](#) objects to the database

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddClusters (
    Euresys.Open_eVision_2_10.EOCR2CharacterCluster[] clusters
)
```

Parameters

clusters

The vector of [EOCR2CharacterCluster](#) objects to be added to the database.

EOCR2CharacterDatabase.Characters

Returns all character from the database.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EOCR2DatabaseCharacter[] Characters
{ get; }
```

EOCR2CharacterDatabase.ClearDatabase

Clears the database.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ClearDatabase(
)
```

EOCR2CharacterDatabase.ClusterDatabase

Performs a clustering on the database.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EOCR2CharacterCluster[] ClusterDatabase(
    int nClusters
)
```

Parameters

nClusters

The amount of clusters to be generated.

EOCR2CharacterDatabase.EOCR2CharacterDatabase

Constructs an [EOCR2CharacterDatabase](#) context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EOCR2CharacterDatabase (
)
void EOCR2CharacterDatabase (
    Euresys.Open_eVision_2_10.EOCR2CharacterDatabase other
)
```

Parameters

other

[EOCR2CharacterDatabase](#) object to be copied.

EOCR2CharacterDatabase.GetCharacter

Returns a character from the database.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EOCR2DatabaseCharacter GetCharacter (
    int index
)
```

Parameters

index

The index of the character to be returned.

EOCR2CharacterDatabase.operator=

Copies all the data from another [EOCR2CharacterDatabase](#) object into the current [EOCR2CharacterDatabase](#) object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EOCR2CharacterDatabase operator=(
    Euresys.Open_eVision_2_10.EOCR2CharacterDatabase other
)
```

Parameters

other

[EOCR2CharacterDatabase](#) object to be copied

EOCR2CharacterDatabase.RemoveCharacter

Removes a character from the database.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void RemoveCharacter (
    int index
)
```

Parameters

index

The index of the character to be removed.

EOCR2CharacterDatabase.Save

Saves the character database to disk.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Save(
    string path
)
```

Parameters

path

The path of the file.

4.129. EOCR2DatabaseCharacter Class

Holds all information related to a database character.

Namespace: Euresys.Open_eVision_2_10

Properties

Bitmap

The bitmap associated to this character.

CharCode

M The ascii code associated to this character.

e

thods

EOCR2DatabaseCharacter

Constructs an [EOCR2DatabaseCharacter](#) context.

operator=

E Copies all the data from another [EOCR2DatabaseCharacter](#) object into the current [EOCR2DatabaseCharacter](#) object

O

CR2DatabaseCharacter.Bitmap

The bitmap associated to this character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EImageBW8 Bitmap  
{ get; }
```

EOCR2DatabaseCharacter.CharacterCode

The ascii code associated to this character.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
ushort CharacterCode
{ get; set; }
```

EOCR2DatabaseCharacter.EOCR2DatabaseCharacter

Constructs an [EOCR2DatabaseCharacter](#) context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EOCR2DatabaseCharacter (
)
void EOCR2DatabaseCharacter (
    Euresys.Open_eVision_2_10.EOCR2DatabaseCharacter other
)
```

Parameters

other

[EOCR2DatabaseCharacter](#) object to be copied.

EOCR2DatabaseCharacter.operator=

Copies all the data from another [EOCR2DatabaseCharacter](#) object into the current [EOCR2DatabaseCharacter](#) object

Namespace: Euresys.Open_eVision_2_10

[C#]

```
Euresys.Open_eVision_2_10.EOCR2DatabaseCharacter operator=(  
    Euresys.Open_eVision_2_10.EOCR2DatabaseCharacter other  
)
```

Parameters

other

[EOCR2DatabaseCharacter](#) object to be copied

4.130. EOCR2Line Class

Holds a vector of [EOCR2Word](#) objects representing a line.

Namespace: Euresys.Open_eVision_2_10

Properties

[BoundingBox](#)

The bounding box encapsulating all characters in the line.

[Text](#)

The true value of all characters in the line.

[Words](#)

M The vector of [EOCR2Word](#) objects representing the line.

e

Methods

[EOCR2Line](#)

Constructs an [EOCR2Line](#) context.

operator=

⌈ Copies all the data from another [EOCR2Line](#) object into the current [EOCR2Line](#) object

CR2Line.BoundingBox

The bounding box encapsulating all characters in the line.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.ERectangle BoundingBox  
{ get; }
```

EOCR2Line.EOCR2Line

Constructs an [EOCR2Line](#) context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void EOCR2Line (  
)  
  
void EOCR2Line (  
    Euresys.Open_eVision_2_10.EOCR2Line other  
)
```

Parameters

other

[EOCR2Line](#) object to be copied.

EOCR2Line.operator=

Copies all the data from another [EOCR2Line](#) object into the current [EOCR2Line](#) object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EOCR2Line operator=(
    Euresys.Open_eVision_2_10.EOCR2Line other
)
```

Parameters

other

[EOCR2Line](#) object to be copied

EOCR2Line.Text

The true value of all characters in the line.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
string Text
{ get; set; }
```

Remarks

Use a space to separate two words.

EOCR2Line.Words

The vector of [EOCR2Word](#) objects representing the line.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EOCR2Word[] Words
{ get; set; }
```

4.131. EOCR2Text Class

Holds a vector of [EOCR2Line](#) objects representing a text.

Namespace: Euresys.Open_eVision_2_10

Properties

[BoundingBox](#)

The bounding box encapsulating all characters in the text.

[Lines](#)

The vector of [EOCR2Line](#) objects representing the text.

[Text](#)

The true value of all characters in the text.

Methods

EOCR2Text

Constructs an [EOCR2Text](#) context.

operator=

Copies all the data from another [EOCR2Text](#) object into the current [EOCR2Text](#) object

○

CR2Text.BoundingBox

The bounding box encapsulating all characters in the text.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.ERectangle BoundingBox  
    { get; }
```

EOCR2Text.EOCR2Text

Constructs an [EOCR2Text](#) context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void EOcr2Text(  
    )  
  
void EOcr2Text(  
    Euresys.Open_eVision_2_10.EOcr2Text other  
    )
```

Parameters

other

[EOcr2Text](#) object to be copied.

Remarks

Default and copy constructors.

EOcr2Text.Lines

The vector of [EOcr2Line](#) objects representing the text.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
Euresys.Open_eVision_2_10.EOcr2Line[] Lines  
  
{ get; set; }
```

EOcr2Text.operator=

Copies all the data from another [EOcr2Text](#) object into the current [EOcr2Text](#) object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EOCR2Text operator=(
    Euresys.Open_eVision_2_10.EOCR2Text other
)
```

Parameters

other

[EOCR2Text](#) object to be copied

EOCR2Text.Text

The true value of all characters in the text.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
string Text
{ get; set; }
```

Remarks

Use a space (" ") to separate two words and a linebreak ("\n") to separate two lines.

4.132. EOCR2Word Class

Holds a vector of [EOCR2Char](#) objects representing a word.

Namespace: Euresys.Open_eVision_2_10

Properties

BoundingBox

The bounding box of the word, encapsulating all characters in the word.

Characters

The vector of [EOCR2Char](#) objects representing the word.

Text

M The true value of all characters in the word.

e

Methods

EOCR2Word

Constructs an [EOCR2Word](#) context.

operator=

E Copies all the data from another [EOCR2Word](#) object into the current [EOCR2Word](#) object

O

CR2Word.BoundingBox

The bounding box of the word, encapsulating all characters in the word.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.ERectangle BoundingBox
```

```
{ get; }
```

EOCR2Word.Characters

The vector of [EOCR2Char](#) objects representing the word.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EOCR2Char[] Characters  
{ get; set; }
```

EOCR2Word.EOCR2Word

Constructs an [EOCR2Word](#) context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void EOCR2Word(  
)  
void EOCR2Word(  
    Euresys.Open_eVision_2_10.EOCR2Word other  
)
```

Parameters

other

[EOCR2Word](#) object to be copied.

EOCR2Word.operator=

Copies all the data from another [EOCR2Word](#) object into the current [EOCR2Word](#) object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EOCR2Word operator=(
    Euresys.Open_eVision_2_10.EOCR2Word other
)
```

Parameters

other

[EOCR2Word](#) object to be copied

EOCR2Word.Text

The true value of all characters in the word.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
string Text
{ get; set; }
```

4.133. EOCV Class

Manages a complete context for the optical character verification tool implemented in EasyOCV.

Namespace: Euresys.Open_eVision_2_10

Properties

AccurateTextsLocationScores

Current texts location scoring mode.

ContrastAverage

Average contrast of last images added to statistics, or **FLOAT32_UNDEFINED** if it cannot be computed.

ContrastDeviation

Standard deviation of the contrast of the last images added to statistics, or **FLOAT32_UNDEFINED** if it cannot be computed.

ContrastTolerance

Allowed tolerance on the global contrast (allowed difference between the sample and template values).

Diagnostics

Logical combination (bitwise OR) of the defects found.

FreeCharCount

Return the number of free characters of the OCV context.

LocationMode

Kind of pre-processing should be applied to the sample image, as defined by [ELocationMode](#).

NormalizeLocationScore

Current location score normalization mode.

NumTexts

Number of texts in the current OCV context.

ReduceLocationScore

Current location score reduction mode.

ResampleChars

Character resampling mode.

SampleBackground

Reference background gray levels determined during inspection of the sample.

SampleContrast

Global contrast of the last inspected image, or **FLOAT32_UNDEFINED** if it has not been computed.

SampleForeground

Reference foreground gray levels determined during inspection of the sample.

SampleThreshold

Threshold level applied to the sample during inspection.

StatisticsCount

Number of samples that have been taken into account in the statistics so far.

TemplateBackground

Reference background gray levels determined during learning of the template.

TemplateContrast

Global contrast of the template image, or **FLOAT32_UNDEFINED** if it has not been computed.

TemplateForeground

Reference foreground gray levels determined during learning of the template.

TemplateImage

Source template image associated to the OCV context during model edition.

TemplateThreshold

Threshold level applied to the template during learning.

UsedQualityIndicators

Choice of quality indicators.

WhiteOnBlack

M Flag indicating whether the learning and inspection steps use white characters on a black background, or black characters on a white background.

thods

AdjustCharsLocationRanges

Adjusts the location ranges (i.e tolerances and bias) of the characters using the results of the statistical training (minimum and maximum observed values).

AdjustCharsQualityRanges

Adjusts the allowed ranges for the character quality indicators (i.e. tolerances and bias) using the results of the statistical training (average and standard deviation of the observed values).

AdjustShiftTolerances

Adjust the shift tolerances on texts from a provided ROI.

AdjustTextsLocationRanges

Adjusts the location ranges (i.e tolerances and bias) of texts using the results of the statistical training (minimum and maximum observed values).

AdjustTextsQualityRanges

Adjusts the allowed ranges for the text quality indicators (i.e. tolerances and bias) using the results of the statistical training (average and standard deviation of the observed values).

ClearStatistics

Resets the statistics accumulation.

ComputeDefaultTolerances

Computes or recomputes the default tolerances of the quality indicators currently in use for all characters of all texts from the given ROI and threshold.

CreateTemplateChars

Adds to the OCV context the characters formed from the list of free objects.

CreateTemplateObjects

Adds to the OCV context the objects from the list of the associated coded image.

CreateTemplateTexts

Adds to the OCV context a piece of text formed from the list of free characters.

DeleteTemplateChars

Removes free characters from the OCV context.

DeleteTemplateObjects

Removes free objects from the OCV context.

DeleteTemplateTexts

Removes template texts from the OCV context.

DrawTemplateChars

Draws the free characters as bounding rectangles.

DrawTemplateCharsWithCurrentPen

Draws the free characters as bounding rectangles.

DrawTemplateObjects

Draws the free objects as blobs.

DrawTemplateObjectsWithCurrentPen

Draws the free objects as blobs.

DrawTemplateTexts

Draws the texts as bounding rectangles.

DrawTemplateTextsChars

Draws the characters of the texts as bounding rectangles.

DrawTemplateTextsCharsWithCurrentPen

Draws the characters of the texts as bounding rectangles.

DrawTemplateTextsWithCurrentPen

Draws the texts as bounding rectangles.

DrawText

Draws a tight bounding box around a single text, possibly with the main diagonal where diagnostics occur.

DrawTextChars

Draws a tight bounding box around all characters of a single text, possibly with the main diagonal where diagnostics occur.

DrawTextCharsWithCurrentPen

Draws a tight bounding box around all characters of a single text, possibly with the main diagonal where diagnostics occur.

DrawTexts

Draws a tight bounding box around all texts, possibly with the main diagonal where diagnostics occur.

DrawTextsChars

Draws a tight bounding box around all characters of all texts.

DrawTextsCharsWithCurrentPen

Draws a tight bounding box around all characters of all texts.

DrawTextsWithCurrentPen

Draws a tight bounding box around all texts, possibly with the main diagonal where diagnostics occur.

DrawTextWithCurrentPen

Draws a tight bounding box around a single text, possibly with the main diagonal where diagnostics occur.

EOCV

Constructs an OCV context.

GatherTextsCharsParameters

Copies the parameters from the characters specified by the selection modes to the temporary character.

GatherTextsParameters

Copies the parameters from the texts specified by the selection mode to the temporary text.

GetFreeChar

Gets an individual free character of the OCV context.

[GetNumTextChars](#)

Returns the number of characters in this OCV context.

[GetText](#)

Returns an individual text of the OCV context.

[GetTextCharParameters](#)

Copies the parameters from the character of given index to the temporary character.

[GetTextCharPoint](#)

Computes the coordinates of a point located relatively to a given character.

[GetTextParameters](#)

Copies the parameters from the text of given index to the temporary text.

[GetTextPoint](#)

Computes the coordinates of a point located relatively to a given text.

[Inspect](#)

Inspection is the process that locates the template in the sample image using all allowed degrees of freedom, compares both images to detects and quantify defects.

[Learn](#)

Pre-processes the model so that all required internal data structures are set up.

[Load](#)

Loads the [EOCV](#). The given [ESerializer](#) must have been created for reading.

Save	Saves the EOCV . The given ESerializer must have been created for writing.
ScatterTextsCharsParameters	Copies the desired parameters from the temporary character to the characters specified by the selection mode.
ScatterTextsParameters	Copies the desired parameters from the temporary text to the texts specified by the selection mode.
SelectSampleTexts	Toggles selection of the template texts whose bounding box intersects a given rectangle.
SelectSampleTextsChars	Toggles selection of the template text characters whose bounding box intersects a given rectangle.
SelectTemplateChars	Toggles selection of the free characters whose bounding box intersects a given rectangle.
SelectTemplateObjects	Toggles selection of the objects whose bounding box intersects a given rectangle.
SelectTemplateTexts	Toggles selection of the template texts whose bounding box intersects a given rectangle.
SetFreeChar	Sets an individual free character of the OCV context.

SetText

Modifies the individual text at the given index by the given value.

SetTextCharParameters

Copies the desired parameters from the temporary character to the character of given index.

SetTextParameters

Copies the desired parameters from the temporary text to the text of given index.

UpdateStatistics

⌊ Adds the parameters of the last inspection to the statistics.

○

CV.AccurateTextsLocationScores

Current texts location scoring mode.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool AccurateTextsLocationScores  
{ get; set; }
```

Remarks

During text location, the text location score is computed before any individual character displacement is allowed. This results in lower location scores because the matching between undeformed template and sample texts is less accurate. When turned on, this property allows the score to be corrected, by taking into account the character displacements. This allows more robust diagnostics based on the location score. If no movement freedom is given to the characters (no **ShiftTolerance**), this option is irrelevant.

EOCV.AdjustCharsLocationRanges

Adjusts the location ranges (i.e tolerances and bias) of the characters using the results of the statistical training (minimum and maximum observed values).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AdjustCharsLocationRanges (
    float factor,
    Euresys.Open_eVision_2_10.ESelectionFlag textSelection,
    Euresys.Open_eVision_2_10.ESelectionFlag charSelection
)
```

Parameters

factor

Safety factor to use when re-computing the tolerances.

textSelection

Tells on which texts the adjustment should be applied, as defined by [ESelectionFlag](#).

charSelection

Tells on which chars the adjustment should be applied, as defined by [ESelectionFlag](#).

Remarks

The method reassigns bias and tolerances values according to their minimum and maximum observed values in the X and Y directions (reported by statistics). The following adjustments are made on characters whose selection state matches **textSelection** and **charSelection**: * **Shift Bias** is assigned $1/2 * (\text{Shift Max} + \text{Shift Min})$ * **Shift Tolerance** is assigned $1/2 * \text{factor} * (\text{Shift Max} - \text{Shift Min})$ Defined this way, the allowed range is simply the observed range enlarged by the user-defined factor. The default value for **factor** is **1.2** (+20 % allowance).

EOCV.AdjustCharsQualityRanges

Adjusts the allowed ranges for the character quality indicators (i.e. tolerances and bias) using the results of the statistical training (average and standard deviation of the observed values).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AdjustCharsQualityRanges (
    float factor,
    Euresys.Open_eVision_2_10.ESelectionFlag textSelection,
    Euresys.Open_eVision_2_10.ESelectionFlag charSelection
)
```

Parameters

factor

Safety factor to use when re-computing the tolerances.

textSelection

Tells on which texts the adjustment should be applied, as defined by [ESelectionFlag](#).

charSelection

Tells on which chars the adjustment should be applied, as defined by [ESelectionFlag](#).

Remarks

The method reassigns bias and tolerances values according to their average and standard deviation, as reported by statistics. The following adjustments are made on characters whose selection state matches **textSelection** and **charSelection**: * the bias parameter value is assigned the corresponding parameter average * the tolerance parameter value is assigned the product of **factor** by its corresponding standard deviation Please note that **3** is a minimum value for **factor** (3sigma rule). This value should be increased if deviations are computed from a small number of samples.

EOCV.AdjustShiftTolerances

Adjust the shift tolerances on texts from a provided ROI.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AdjustShiftTolerances (
    Euresys.Open_eVision_2_10.EROIBW8 roi
)
```

Parameters

roi

Pointer on the ROI from which the shift tolerances will be computed.

Remarks

The method performs the computation of texts shift tolerances, assuming that the specified ROI always contains the whole text (text cannot lie outside of the ROI). Other text location parameters (skew, scale, shear) are not taken in account, and this method should not be called if these parameters tolerances are not set to zero. The method also assumes texts always have the same displacement, and does not affect characters location tolerances.

EOCV.AdjustTextsLocationRanges

Adjusts the location ranges (i.e tolerances and bias) of texts using the results of the statistical training (minimum and maximum observed values).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AdjustTextsLocationRanges (
    float factor,
    Euresys.Open_eVision_2_10.ESelectionFlag selection
)
```

Parameters

factor

Safety factor to use when re-computing the tolerances.

selection

Tells on which texts the adjustment should be applied, as defined by [ESelectionFlag](#).

Remarks

The method reassigns bias and tolerances values according to their minimum and maximum observed values in the X and Y directions (reported by statistics). The following adjustments are made on texts whose selection state matches **selection**: * **Shift Bias** is assigned $1/2 * (\text{Shift Max} + \text{Shift Min})$ * **Shift Tolerance** is assigned $1/2 * \text{factor} * (\text{Shift Max} - \text{Shift Min})$ Defined this way, the allowed range is simply the observed range enlarged by the user-defined factor. The default value for **factor** is **1.2** (+20 % allowance).

EOCV.AdjustTextsQualityRanges

Adjusts the allowed ranges for the text quality indicators (i.e. tolerances and bias) using the results of the statistical training (average and standard deviation of the observed values).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AdjustTextsQualityRanges (
    float factor,
    Euresys.Open_eVision_2_10.ESelectionFlag selection
)
```

Parameters

factor

Safety factor to use when re-computing the tolerances.

selection

Tells on which texts the adjustment should be applied, as defined by [ESelectionFlag](#).

Remarks

The method reassigns bias and tolerances values according to their average and standard deviation, as reported by statistics. The following adjustments are made on texts whose selection state matches **selection**: * the bias parameter value is assigned the corresponding parameter average * the tolerance parameter value is assigned the product of **factor** by its corresponding standard deviation Please note that **3** is a minimum value for **factor** (3sigma rule). This value should be increased if deviations are computed from a small number of samples.

EOCV.ClearStatistics

Resets the statistics accumulation.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
void ClearStatistics (
)
```

Remarks

This function must be called before a batch of inspections.

EOCV.ComputeDefaultTolerances

Computes or recomputes the default tolerances of the quality indicators currently in use for all characters of all texts from the given ROI and threshold.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ComputeDefaultTolerances (
    Euresys.Open_eVision_2_10.EROIBW8 roi,
    uint threshold
)
```

Parameters

roi

Pointer to the ROI from which the default tolerances should be computed.

threshold

Threshold level to use during computation, as defined by [EThresholdMode](#).

Remarks

An explicit call of this method is performed by method `Learn` at the end of the learning stage.

EOCV.ContrastAverage

Average contrast of last images added to statistics, or **FLOAT32_UNDEFINED** if it cannot be computed.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ContrastAverage  
  
    { get; }
```

Remarks

The contrast is defined as the ratio of the reference gray-levels difference over their sum. This parameter is maximum (100 %) for a perfectly contrasted image (white on black).

EOCV.ContrastDeviation

Standard deviation of the contrast of the last images added to statistics, or **FLOAT32_UNDEFINED** if it cannot be computed.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ContrastDeviation  
  
    { get; }
```

Remarks

The contrast is defined as the ratio of the reference gray-levels difference over their sum. This parameter is maximum (100 %) for a perfectly contrasted image (white on black).

EOCV.ContrastTolerance

Allowed tolerance on the global contrast (allowed difference between the sample and template values).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ContrastTolerance  
  
{ get; set; }
```

Remarks

The contrast is defined as the ratio of the reference gray-levels difference over their sum. This parameter is maximum (100 %) for a perfectly contrasted image (white on black).

EOCV.CreateTemplateChars

Adds to the OCV context the characters formed from the list of free objects.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void CreateTemplateChars (  
    Euresys.Open_eVision_2_10.ESelectionFlag objectsSelectionFlag,  
    Euresys.Open_eVision_2_10.ECharCreationMode creationMode  
)
```

Parameters

objectsSelectionFlag

Selection mode for the objects, as defined by [ESelectionFlag](#). By default, only the selected objects are created.

creationMode

Free objects grouping criterion, as defined by [ECharCreationMode](#). By default, the free objects whose bounding rectangle overlap are considered as belonging to the same character.

Remarks

The free characters are sets of blobs taken from the free objects list. One step of the model definition is to specify which free objects will be handled by the OCV context and how they will be grouped to form characters. Grouping can be done automatically by using an overlapping criterion, or explicitly. When an object is added to a character, it is removed from the free objects list.

EOCV.CreateTemplateObjects

Adds to the OCV context the objects from the list of the associated coded image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void CreateTemplateObjects (
    Euresys.Open_eVision_2_10.ECodedImage codedImage,
    Euresys.Open_eVision_2_10.ESelectionFlag codedObjectsSelectionFlag
)
```

Parameters

codedImage

An **ECodedImage** object.

codedObjectsSelectionFlag

Selection mode for the objects, as defined by [ESelectionFlag](#). By default, only the selected objects are created.

Remarks

The free objects are blobs built using a separate coded image. One step of the model definition is to specify which objects will be handled by the OCV context.

EOCV.CreateTemplateTexts

Adds to the OCV context a piece of text formed from the list of free characters.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void CreateTemplateTexts (  
    Euresys.Open_eVision_2_10.ESelectionFlag charsSelectionFlag  
)
```

Parameters

charsSelectionFlag

Selection mode for the free characters, as defined by [ESelectionFlag](#). By default, only the selected free characters are processed.

Remarks

The template texts are sets of characters taken from the free characters list. One step of the model definition is to specify which free characters will be handled by the OCV context and how they will be grouped to form texts. Grouping must be done explicitly. When a character is added to a text, it is removed from the free characters list.

EOCV.DeleteTemplateChars

Removes free characters from the OCV context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void DeleteTemplateChars (  
    Euresys.Open_eVision_2_10.ESelectionFlag charsSelectionFlag  
)
```

Parameters

charsSelectionFlag

Selection mode for the free characters, as defined by [ESelectionFlag](#). By default, only the selected free characters are deleted.

Remarks

The free characters are sets of blobs taken from the free objects list. One step of the model definition is to specify which free objects will be handled by the OCV context and how they will be grouped to form characters. Grouping can be done automatically by using a overlapping criterion, or explicitly. When an object is added to a character, it is removed from the free objects list.

EOCV.DeleteTemplateObjects

Removes free objects from the OCV context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DeleteTemplateObjects (
    Euresys.Open_eVision_2_10.ESelectionFlag objectsSelectionFlag
)
```

Parameters

objectsSelectionFlag

Selection mode for the objects, as defined by [ESelectionFlag](#). By default, only the selected objects are deleted.

Remarks

The free objects are blobs built using a separate coded image. One step of the model definition is to specify which objects will be handled by the OCV context.

EOCV.DeleteTemplateTexts

Removes template texts from the OCV context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DeleteTemplateTexts (
    Euresys.Open_eVision_2_10.ESelectionFlag textsSelectionFlag
)
```

Parameters

textsSelectionFlag

Selection mode for the template texts, as defined by [ESelectionFlag](#). By default, only the selected texts are deleted.

Remarks

The template texts are sets of characters taken from the free characters list. One step of the model definition is to specify which free characters will be handled by the OCV context and how they will be grouped to form texts. Grouping must be done explicitly. When a character is added to a text, it is removed from the free characters list.

EOCV.Diagnostics

Logical combination (bitwise OR) of the defects found.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
uint Diagnostics  
    { get; }
```

Remarks

Inspection is the process that locates the template in the sample image, using all allowed degrees of freedom, and that compares both images to detect and quantify defects. After inspection, all defective texts and text characters are selected. Further, the diagnostics binary mask contain a detailed interpretation of the defects found.

EOCV.DrawTemplateChars

Draws the free characters as bounding rectangles.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```

void DrawTemplateChars (
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ESelectionFlag charsSelectionFlag,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)

void DrawTemplateChars (
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.ESelectionFlag charsSelectionFlag,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)

void DrawTemplateChars (
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.ESelectionFlag charsSelectionFlag,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)

```

Parameters

graphicContext

Device context of the drawing window.

charsSelectionFlag

Selection mode for the free characters, as defined by [ESelectionFlag](#). By default, only the selected characters are drawn.

zoomX

Zooming parameters.

zoomY

Zooming parameters.

originX

Panning parameters.

originY

Panning parameters.

color

-

Remarks

The template drawing methods are used to graphically represent the free objects, free characters, texts and text characters associated to the OCV context before learning. All drawing operations are done using the current pen attributes.

EOCV.DrawTemplateCharsWithCurrentPen

Draws the free characters as bounding rectangles.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void DrawTemplateCharsWithCurrentPen (  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ESelectionFlag charsSelectionFlag,  
    float zoomX,  
    float zoomY,  
    float originX,  
    float originY  
)
```

Parameters

graphicContext

Device context of the drawing window.

charsSelectionFlag

Selection mode for the free characters, as defined by [ESelectionFlag](#). By default, only the selected characters are drawn.

zoomX

Zooming parameters.

zoomY

Zooming parameters.

originX

Panning parameters.

originY

Panning parameters.

Remarks

The template drawing methods are used to graphically represent the free objects, free characters, texts and text characters associated to the OCV context before learning. All drawing operations are done using the current pen attributes.

EOCV.DrawTemplateObjects

Draws the free objects as blobs.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void DrawTemplateObjects(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ESelectionFlag objectsSelectionFlag,  
    float zoomX,  
    float zoomY,  
    float originX,  
    float originY  
)
```

```
void DrawTemplateObjects(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ERGBColor color,  
    Euresys.Open_eVision_2_10.ESelectionFlag objectsSelectionFlag,  
    float zoomX,  
    float zoomY,  
    float originX,  
    float originY  
)
```

```
void DrawTemplateObjects(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.ESelectionFlag objectsSelectionFlag,  
    float zoomX,  
    float zoomY,  
    float originX,  
    float originY  
)
```

Parameters

graphicContext

Device context of the drawing window.

objectsSelectionFlag

Selection mode for the objects, as defined by [ESelectionFlag](#). By default, only the selected objects are drawn.

zoomX

Zooming parameters.

zoomY

Zooming parameters.

originX

Panning parameters.

originY

Panning parameters.

color

The color in which to draw the overlay.

Remarks

The associated coded image must be present. The template drawing methods are used to graphically represent the free objects, free characters, texts and text characters associated to the OCV context before learning. All drawing operations are done using the current pen attributes.

EOCV.DrawTemplateObjectsWithCurrentPen

Draws the free objects as blobs.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawTemplateObjectsWithCurrentPen (
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ESelectionFlag objectsSelectionFlag,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)
```

Parameters

graphicContext

Device context of the drawing window.

objectsSelectionFlag

Selection mode for the objects, as defined by [ESelectionFlag](#). By default, only the selected objects are drawn.

zoomX

Zooming parameters.

zoomY

Zooming parameters.

originX

Panning parameters.

originY

Panning parameters.

Remarks

The associated coded image must be present. The template drawing methods are used to graphically represent the free objects, free characters, texts and text characters associated to the OCV context before learning. All drawing operations are done using the current pen attributes.

EOCV.DrawTemplateTexts

Draws the texts as bounding rectangles.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawTemplateTexts (
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ESelectionFlag textsSelectionFlag,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)
```

```

void DrawTemplateTexts (
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.ESelectionFlag textsSelectionFlag,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)

void DrawTemplateTexts (
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.ESelectionFlag textsSelectionFlag,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)

```

Parameters

graphicContext

Device context of the drawing window.

textsSelectionFlag

Selection mode for the free texts, as defined by [ESelectionFlag](#). By default, only the selected texts are drawn.

zoomX

Zooming parameters.

zoomY

Zooming parameters.

originX

Panning parameters.

originY

Panning parameters.

color

The color in which to draw the overlay.

Remarks

The template drawing methods are used to graphically represent the free objects, free characters, texts and text characters associated to the OCV context before learning. All drawing operations are done using the current pen attributes.

EOCV.DrawTemplateTextsChars

Draws the characters of the texts as bounding rectangles.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void DrawTemplateTextsChars (
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ESelectionFlag textsSelectionFlag,
    Euresys.Open_eVision_2_10.ESelectionFlag charsSelectionFlag,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)

void DrawTemplateTextsChars (
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.ESelectionFlag textsSelectionFlag,
    Euresys.Open_eVision_2_10.ESelectionFlag charsSelectionFlag,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)

void DrawTemplateTextsChars (
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.ESelectionFlag textsSelectionFlag,
    Euresys.Open_eVision_2_10.ESelectionFlag charsSelectionFlag,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)
```

Parameters

graphicContext

Device context of the drawing window.

textsSelectionFlag

Selection mode for the texts, as defined by [ESelectionFlag](#). By default, only characters from the selected texts are drawn.

charsSelectionFlag

Selection mode for the characters, as defined by [ESelectionFlag](#). By default, only the selected characters of the selected texts are drawn.

zoomX

Zooming parameters.

zoomY

Zooming parameters.

originX

Panning parameters.

originY

Panning parameters.

color

The color in which to draw the overlay.

Remarks

The template drawing methods are used to graphically represent the free objects, free characters, texts and text characters associated to the OCV context before learning. All drawing operations are done using the current pen attributes.

EOCV.DrawTemplateTextsCharsWithCurrentPen

Draws the characters of the texts as bounding rectangles.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void DrawTemplateTextsCharsWithCurrentPen(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ESelectionFlag textsSelectionFlag,  
    Euresys.Open_eVision_2_10.ESelectionFlag charsSelectionFlag,  
    float zoomX,  
    float zoomY,  
    float originX,  
    float originY  
)
```

Parameters

graphicContext

Device context of the drawing window.

textsSelectionFlag

Selection mode for the texts, as defined by [ESelectionFlag](#). By default, only characters from the selected texts are drawn.

charsSelectionFlag

Selection mode for the characters, as defined by [ESelectionFlag](#). By default, only the selected characters of the selected texts are drawn.

zoomX

Zooming parameters.

zoomY

Zooming parameters.

originX

Panning parameters.

originY

Panning parameters.

Remarks

The template drawing methods are used to graphically represent the free objects, free characters, texts and text characters associated to the OCV context before learning. All drawing operations are done using the current pen attributes.

EOCV.DrawTemplateTextsWithCurrentPen

Draws the texts as bounding rectangles.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawTemplateTextsWithCurrentPen(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ESelectionFlag textsSelectionFlag,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)
```

Parameters

graphicContext

Device context of the drawing window.

textsSelectionFlag

Selection mode for the free texts, as defined by [ESelectionFlag](#). By default, only the selected texts are drawn.

zoomX

Zooming parameters.

zoomY

Zooming parameters.

originX

Panning parameters.

originY

Panning parameters.

Remarks

The template drawing methods are used to graphically represent the free objects, free characters, texts and text characters associated to the OCV context before learning. All drawing operations are done using the current pen attributes.

EOCV.DrawText

Draws a tight bounding box around a single text, possibly with the main diagonal where diagnostics occur.

Namespace: Euresys.Open_eVision_2_10

```

[C#]
void DrawText(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EOCVText text,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)

void DrawText(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.EOCVText text,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)

void DrawText(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.EOCVText text,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)

```

Parameters

graphicContext

Device context of the drawing window.

text

Reference to the desired text can be obtained by using [EOCV::GetText](#).

zoomX

Zooming parameters.

zoomY

Zooming parameters.

originX

Panning parameters.

originY

Panning parameters.

color

The color in which to draw the overlay.

Remarks

The sample drawing member functions are used to graphically represent the texts and text characters located in the image during inspection. All drawing operations are done using the current pen attributes. Normally, the characters and texts are drawn as a bounding box. However, when diagnostics have been raised on a character or text, the main diagonal (north-west to south-east) is drawn as well, indicating that the item is rejected.

EOCV.DrawTextChars

Draws a tight bounding box around all characters of a single text, possibly with the main diagonal where diagnostics occur.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void DrawTextChars (
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EOCVText text,
    Euresys.Open_eVision_2_10.ESelectionFlag charsSelectionFlag,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)

void DrawTextChars (
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.EOCVText text,
    Euresys.Open_eVision_2_10.ESelectionFlag charsSelectionFlag,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)
```

```
void DrawTextChars (  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EOCVText text,  
    Euresys.Open_eVision_2_10.ESelectionFlag charsSelectionFlag,  
    float zoomX,  
    float zoomY,  
    float originX,  
    float originY  
)
```

Parameters

graphicContext

Handle to the device context of the destination window.

text

Reference to the desired text.

charsSelectionFlag

Characters selection mode, as defined by [ESelectionFlag](#).

zoomX

Zooming parameters.

zoomY

Zooming parameters.

originX

Panning parameters.

originY

Panning parameters.

color

The color in which to draw the overlay.

Remarks

The sample drawing methods are used to graphically represent the texts and text characters located in the image during inspection. All drawing operations are done using the current pen attributes. Normally, the characters and texts are drawn as a bounding box. However, when diagnostics have been raised on a character or text, the main diagonal (north west to south east) is drawn as well, indicating that the item is rejected.

EOCV.DrawTextCharsWithCurrentPen

Draws a tight bounding box around all characters of a single text, possibly with the main diagonal where diagnostics occur.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawTextCharsWithCurrentPen (
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EOCVText text,
    Euresys.Open_eVision_2_10.ESelectionFlag charsSelectionFlag,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)
```

Parameters

graphicContext

Handle to the device context of the destination window.

text

Reference to the desired text.

charsSelectionFlag

Characters selection mode, as defined by [ESelectionFlag](#).

zoomX

Zooming parameters.

zoomY

Zooming parameters.

originX

Panning parameters.

originY

Panning parameters.

Remarks

The sample drawing methods are used to graphically represent the texts and text characters located in the image during inspection. All drawing operations are done using the current pen attributes. Normally, the characters and texts are drawn as a bounding box. However, when diagnostics have been raised on a character or text, the main diagonal (north west to south east) is drawn as well, indicating that the item is rejected.

EOCV.DrawTexts

Draws a tight bounding box around all texts, possibly with the main diagonal where diagnostics occur.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void DrawTexts(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ESelectionFlag textsSelectionFlag,  
    float zoomX,  
    float zoomY,  
    float originX,  
    float originY  
)  
  
void DrawTexts(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ERGBColor color,  
    Euresys.Open_eVision_2_10.ESelectionFlag textsSelectionFlag,  
    float zoomX,  
    float zoomY,  
    float originX,  
    float originY  
)  
  
void DrawTexts(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.ESelectionFlag textsSelectionFlag,  
    float zoomX,  
    float zoomY,  
    float originX,  
    float originY  
)
```

Parameters

graphicContext

Device context of the drawing window.

textsSelectionFlag

Texts selection mode, as defined by [ESelectionFlag](#).

zoomX

Zooming parameters.

zoomY

Zooming parameters.

originX

Panning parameters.

originY

Panning parameters.

color

The color in which to draw the overlay.

Remarks

The sample drawing methods are used to graphically represent the texts and text characters located in the image during inspection. All drawing operations are done using the current pen attributes. Normally, the characters and texts are drawn as a bounding box. However, when diagnostics have been raised on a character or text, the main diagonal (north west to south east) is drawn as well, indicating that the item is rejected.

EOCV.DrawTextsChars

Draws a tight bounding box around all characters of all texts.

Namespace: Euresys.Open_eVision_2_10

[C#]

```

void DrawTextsChars (
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ESelectionFlag textsSelectionFlag,
    Euresys.Open_eVision_2_10.ESelectionFlag charsSelectionFlag,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)

void DrawTextsChars (
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.ESelectionFlag textsSelectionFlag,
    Euresys.Open_eVision_2_10.ESelectionFlag charsSelectionFlag,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)

void DrawTextsChars (
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.ESelectionFlag textsSelectionFlag,
    Euresys.Open_eVision_2_10.ESelectionFlag charsSelectionFlag,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)

```

Parameters

graphicContext

Device context of the drawing window.

textsSelectionFlag

Texts selection mode, as defined by [ESelectionFlag](#).

charsSelectionFlag

Characters selection mode, as defined by [ESelectionFlag](#).

zoomX

Zooming parameters.

zoomY

Zooming parameters.

originX

Panning parameters.

originY

Panning parameters.

color

The color in which to draw the overlay.

Remarks

The sample drawing methods are used to graphically represent the texts and text characters located in the image during inspection. All drawing operations are done using the current pen attributes. Normally, the characters and texts are drawn as a bounding box. However, when diagnostics have been raised on a character or text, the main diagonal (north west to south east) is drawn as well, indicating that the item is rejected.

EOCV.DrawTextsCharsWithCurrentPen

Draws a tight bounding box around all characters of all texts.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawTextsCharsWithCurrentPen (
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ESelectionFlag textsSelectionFlag,
    Euresys.Open_eVision_2_10.ESelectionFlag charsSelectionFlag,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)
```

Parameters

graphicContext

Device context of the drawing window.

textsSelectionFlag

Texts selection mode, as defined by [ESelectionFlag](#).

charsSelectionFlag

Characters selection mode, as defined by [ESelectionFlag](#).

zoomX

Zooming parameters.

zoomY

Zooming parameters.

originX

Panning parameters.

originY

Panning parameters.

Remarks

The sample drawing methods are used to graphically represent the texts and text characters located in the image during inspection. All drawing operations are done using the current pen attributes. Normally, the characters and texts are drawn as a bounding box. However, when diagnostics have been raised on a character or text, the main diagonal (north west to south east) is drawn as well, indicating that the item is rejected.

EOCV.DrawTextsWithCurrentPen

Draws a tight bounding box around all texts, possibly with the main diagonal where diagnostics occur.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawTextsWithCurrentPen(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ESelectionFlag textsSelectionFlag,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)
```

Parameters

graphicContext

Device context of the drawing window.

textsSelectionFlag

Texts selection mode, as defined by [ESelectionFlag](#).

zoomX

Zooming parameters.

zoomY

Zooming parameters.

originX

Panning parameters.

originY

Panning parameters.

Remarks

The sample drawing methods are used to graphically represent the texts and text characters located in the image during inspection. All drawing operations are done using the current pen attributes. Normally, the characters and texts are drawn as a bounding box. However, when diagnostics have been raised on a character or text, the main diagonal (north west to south east) is drawn as well, indicating that the item is rejected.

EOCV.DrawTextWithCurrentPen

Draws a tight bounding box around a single text, possibly with the main diagonal where diagnostics occur.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawTextWithCurrentPen (
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EOCVText text,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)
```

Parameters

graphicContext

Device context of the drawing window.

text

Reference to the desired text can be obtained by using [EOCV::GetText](#).

zoomX

Zooming parameters.

zoomY

Zooming parameters.

originX

Panning parameters.

originY

Panning parameters.

Remarks

The sample drawing member functions are used to graphically represent the texts and text characters located in the image during inspection. All drawing operations are done using the current pen attributes. Normally, the characters and texts are drawn as a bounding box. However, when diagnostics have been raised on a character or text, the main diagonal (north-west to south-east) is drawn as well, indicating that the item is rejected.

EOCV.EOCV

Constructs an OCV context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EOCV(
    Euresys.Open_eVision_2_10.EOCV other
)
void EOCV(
)
```

Parameters

other

-

Remarks

Only the default constructor is needed.

EOCV.FreeCharCount

Return the number of free characters of the OCV context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint FreeCharCount
{ get; }
```

EOCV.GatherTextsCharsParameters

Copies the parameters from the characters specified by the selection modes to the temporary character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GatherTextsCharsParameters (
    Euresys.Open_eVision_2_10.EOCVChar ocvChar,
    Euresys.Open_eVision_2_10.ESelectionFlag textsSelectionFlag,
    Euresys.Open_eVision_2_10.ESelectionFlag charsSelectionFlag
)
```

Parameters

ocvChar

Temporary [EOCVChar](#) object.

textsSelectionFlag

Selection mode for the texts, as defined by [ESelectionFlag](#).

charsSelectionFlag

Selection mode for the text characters, as defined by [ESelectionFlag](#).

Remarks

For each character parameter that has a unique value among the specified characters, the corresponding value is different from undefined. Always be sure to check that a returned parameter does not have such an undefined value assigned, otherwise its contents will appear meaningless.

EOCV.GatherTextsParameters

Copies the parameters from the texts specified by the selection mode to the temporary text.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GatherTextsParameters (
    Euresys.Open_eVision_2_10.EOCVText Text,
    Euresys.Open_eVision_2_10.ESelectionFlag textsSelectionFlag
)
```

Parameters

Text

Selection mode for the texts, as defined by [ESelectionFlag](#).

textsSelectionFlag

Temporary [EOCVText](#) object.

Remarks

For each text parameter that has a unique value among the specified texts, the corresponding value is different from undefined. Always be sure to check that a returned parameter does not have such an undefined value assigned, otherwise its contents will appear meaningless.

EOCV.GetFreeChar

Gets an individual free character of the OCV context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EOCVChar GetFreeChar (
    int index
)
```

Parameters

index

Index of the individual free character.

Remarks

The free characters are sets of blobs taken from the free objects list. One step of the model definition is to specify which free objects will be handled by the OCV context, and how they will be grouped to form characters. Grouping can be done automatically by using a overlapping criterion, or explicitly. When an object is added to a character, it is removed from the free objects list.

EOCV.GetNumTextChars

Returns the number of characters in this OCV context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint GetNumTextChars (
    uint index
)
```

Parameters

index

Index of the desired text, in range **0** to **NumTexts-1**.

Remarks

The characters are numbered from **0** to **NumTextChars** excluded.

EOCV.GetText

Returns an individual text of the OCV context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EOCVText GetText(
    int index
)
```

Parameters

index

Index, between **0** and [EOCV::NumTexts](#) (excluded) of the text to be accessed.

EOCV.GetTextCharParameters

Copies the parameters from the character of given index to the temporary character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetTextCharParameters (
    Euresys.Open_eVision_2_10.EOCVChar ocvChar,
    uint textIndex,
    uint charIndex
)
```

Parameters

ocvChar

Temporary [EOCVChar](#) object.

textIndex

Character index in range **0** to **NumTexts-1**.

charIndex

Character index in range **0** to **NumTextChars-1**.

Remarks

For each character parameter retrieved, the corresponding value is different from undefined. Always be sure to check that a returned parameter does not have such an undefined value assigned, otherwise its contents will appear meaningless.

EOCV.GetTextCharPoint

Computes the coordinates of a point located relatively to a given character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetTextCharPoint(
    uint textIndex,
    uint charIndex,
    out int x,
    out int y,
    float reducedX,
    float reducedY,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

Parameters

textIndex

Text index in range **0..NumTexts**, excluded.

charIndex

Character index in range **0..NumTextChars**, excluded.

x

returned abscissa of the requested point.

y

returned ordinate of the requested point.

reducedX

X position of the requested point relatively to the character bounding box, as defined on the picture below.

reducedY

Y position of the requested point relatively to the character bounding box, as defined on the picture below

zoomX

Zooming parameters.

zoomY

Zooming parameters.

panX

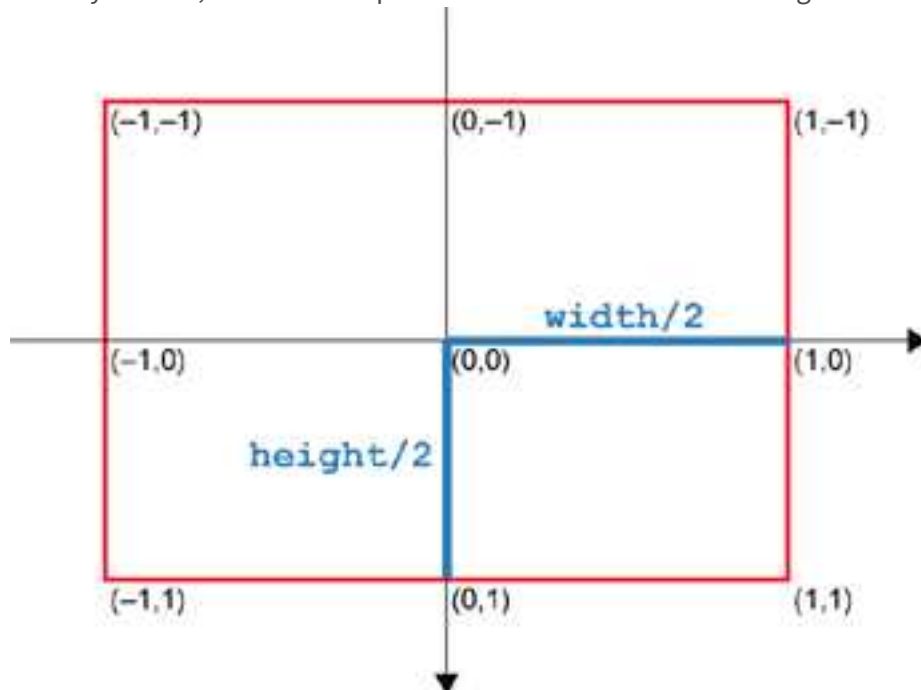
Panning parameters.

panY

Panning parameters.

Remarks

The parameters **reducedX** and **reducedY** are used to indicate the position of the requested point relatively to the bounding box. These parameters can take values from **-1** to **1**, where **1** is the half height or half width of the bounding box. By default, the returned point is the center of the bounding box.



EOCV.GetTextParameters

Copies the parameters from the text of given index to the temporary text.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void GetTextParameters(  
    Euresys.Open_eVision_2_10.EOCVText Text,  
    uint textIndex  
)
```

Parameters

Text

Text index in range **0** to **NumTexts**, excluded.

textIndex

Text index in range **0** to **NumTexts**, excluded.

Remarks

For each text parameter retrieved, the corresponding value is different from undefined. Always be sure to check that a returned parameter does not have such an undefined value assigned, otherwise its contents will appear meaningless.

EOCV.GetTextPoint

Computes the coordinates of a point located relatively to a given text.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void GetTextPoint(  
    uint textIndex,  
    out int x,  
    out int y,  
    float reducedX,  
    float reducedY,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

Parameters

textIndex

Text index in range **0..NumTexts**, excluded.

x

Returned abscissa of the requested point.

y

Returned ordinate of the requested point.

reducedX

X position of the requested point relatively the text bounding box, as defined on the picture below.

reducedY

Y position of the requested point relatively to the text bounding box, as defined on the picture below.

zoomX

Zooming parameters.

zoomY

Zooming parameters.

panX

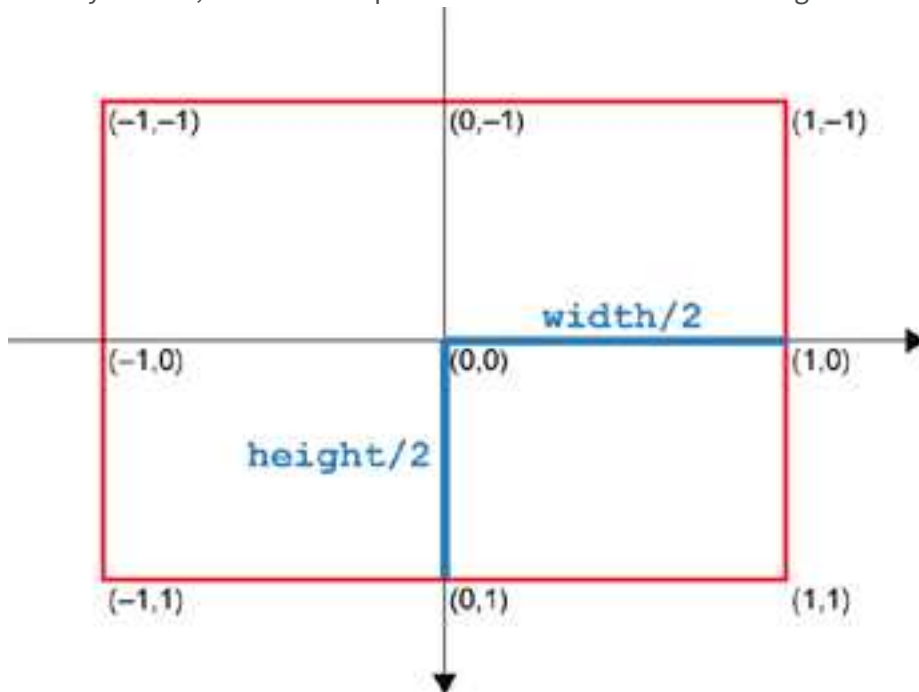
Panning parameters.

panY

Panning parameters.

Remarks

The parameters **reducedX** and **reducedY** are used to indicate the position of the requested point relatively to the bounding box. These parameters can take values from **-1** to **1**, where **1** is the half height or half width of the bounding box. By default, the returned point is the center of the bounding box.



EOCV.Inspect

Inspection is the process that locates the template in the sample image using all allowed degrees of freedom, compares both images to detect and quantify defects.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Inspect(
    Euresys.Open_eVision_2_10.EROIBW8 sampleImage,
    uint threshold
)
```

Parameters

sampleImage

Pointer to the image/ROI to be inspected. The center of this image serves as a reference point on which the center of the template image is aligned.

threshold

Threshold level suitable for the sample image. The same convention as that of **ImgThreshold** is used for automatic thresholding.

Remarks

After inspection, all defective texts and text characters are selected. Further, the Diagnostics binary mask contain a detailed interpretation of the defects found. A few options can be tuned to speed up the inspection process (see Location Options and Inspection Options).

EOCV.Learn

Pre-processes the model so that all required internal data structures are set up.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Learn(
    Euresys.Open_eVision_2_10.EROIBW8 image,
    Euresys.Open_eVision_2_10.ESelectionFlag textsSelectionFlag,
    Euresys.Open_eVision_2_10.ESelectionFlag charsSelectionFlag
)
```

Parameters

image

Pointer to the template image/ROI.

textsSelectionFlag

Texts selection mode, as defined by [ESelectionFlag](#).

charsSelectionFlag

Characters selection mode, as defined by [ESelectionFlag](#).

Remarks

After learning, the model may not be edited anymore. It can be saved for later use and becomes ready for the inspection task. Learning is the final step of the model editing.

EOCV.Load

Loads the [EOCV](#). The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Load(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is read from.

EOCV.LocationMode

Kind of pre-processing should be applied to the sample image, as defined by [ELocationMode](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ELocationMode LocationMode
{ get; set; }
```

Remarks

The location process is an important phase of the inspection task. It allows the OCV context to accurately find the template position in the sample image, using all allowed degrees of freedom. This process can be tuned in several ways to trade speed for robustness.

EOCV.NormalizeLocationScore

Current location score normalization mode.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool NormalizeLocationScore  
    { get; set; }
```

Remarks

Location score normalization performs a score correction, using the sample and template reference gray levels. It is intended to correct potential contrast or intensity discrepancies between template and sample images. It acts as if the sample image had the same foreground and background reference gray levels than the template image. It is recommended to turn this option on if the acquisition conditions can change from one sample to another. However, the correction may have a bad effect if the image acquisition system (camera) uses gamma correction, or the acquired image is saturated. The *location score normalization* mode is independent of the *location score reduction* mode.

EOCV.NumTexts

Number of texts in the current OCV context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint NumTexts  
    { get; }
```

Remarks

The texts are numbered from **0** to **NumTexts-1**.

EOCV.ReduceLocationScore

Current location score reduction mode.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool ReduceLocationScore  
  
{ get; set; }
```

Remarks

The location score is computed as a sum of gray-level values along the character contours. When the reduction mode is enabled (default), the location score is divided by the number of contour points, giving an average gray level, in range **0..255**. The non-reduction mode is considered obsolete. The *location score reduction* mode is independent of the *location score normalization* mode.

EOCV.ResampleChars

Character resampling mode.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool ResampleChars  
  
{ get; set; }
```

Remarks

Normally, when text is inspected with rotation, scaling and/or shearing, some resampling must be performed when computing the quality indicators on the separate characters. If the angles remain small and the scale factors remain very close to unity, this resampling can be avoided by setting this property to **FALSE**.

EOCV.SampleBackground

Reference background gray levels determined during inspection of the sample.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float SampleBackground  
  
{ get; set; }
```

Remarks

Image contrast is an important factor during both learning and inspection. The background and foreground information must be separated using an appropriate threshold, possibly determined automatically. After a threshold is given, the average gray level of the background and foreground areas are computed separately. These serve as reference gray levels to measure the image contrast and normalize the gray level quality indicators. The background and foreground reference gray levels are computed for both the template and sample images. They are used for contrast assessment as well as gray-level normalization.

EOCV.SampleContrast

Global contrast of the last inspected image, or **FLOAT32_UNDEFINED** if it has not been computed.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float SampleContrast  
  
{ get; }
```

Remarks

The contrast is defined as the ratio of the reference gray-levels difference over their sum. This parameter is maximum (100 %) for a perfectly contrasted image (white on black).

EOCV.SampleForeground

Reference foreground gray levels determined during inspection of the sample.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float SampleForeground  
  
    { get; set; }
```

Remarks

Image contrast is an important factor during both learning and inspection. The background and foreground information must be separated using an appropriate threshold, possibly determined automatically. After a threshold is given, the average gray level of the background and foreground areas are computed separately. These serve as reference gray levels to measure the image contrast and normalize the gray level quality indicators. The background and foreground reference gray levels are computed for both the template and sample images. They are used for contrast assessment as well as gray-level normalization.

EOCV.SampleThreshold

Threshold level applied to the sample during inspection.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EBWS SampleThreshold  
  
    { get; set; }
```

Remarks

Image contrast is an important factor during both learning and inspection. The background and foreground information must be separated using an appropriate threshold, possibly determined automatically. After a threshold is given, the average gray level of the background and foreground areas are computed separately. These serve as reference gray levels to measure the image contrast and normalize the gray level quality indicators. The background and foreground reference gray levels are computed for both the template and sample images. They are used for contrast assessment as well as gray-level normalization.

EOCV.Save

Saves the [EOCV](#). The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is written to.

EOCV.ScatterTextsCharsParameters

Copies the desired parameters from the temporary character to the characters specified by the selection mode.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void ScatterTextsCharsParameters (
    Euresys.Open_eVision_2_10.EOCVChar ocvChar,
    Euresys.Open_eVision_2_10.ESelectionFlag textsSelectionFlag,
    Euresys.Open_eVision_2_10.ESelectionFlag charsSelectionFlag
)
```

Parameters

ocvChar

Temporary [EOCVChar](#) object.

textsSelectionFlag

Selection mode for the texts, as defined by [ESelectionFlag](#).

charsSelectionFlag

Selection mode for the text characters, as defined by [ESelectionFlag](#).

Remarks

Only the parameters for which the value is not undefined are copied. Before setting parameters, be sure to call the [EOCVChar::ResetParameters](#) member of the temporary [EOCVChar](#) object. This will reset all parameters to the undefined value, thus avoiding unwanted copy.

EOCV.ScatterTextsParameters

Copies the desired parameters from the temporary text to the texts specified by the selection mode.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ScatterTextsParameters (
    Euresys.Open_eVision_2_10.EOCVText Text,
    Euresys.Open_eVision_2_10.ESelectionFlag textsSelectionFlag
)
```

Parameters

Text

Selection mode for the texts, as defined by [ESelectionFlag](#).

textsSelectionFlag

Temporary [EOCVText](#) object.

Remarks

Only the parameters for which the value is not undefined value are copied. Before setting parameters, be sure to call the [EOCVText::ResetParameters](#) member of the temporary [EOCVText](#) object. This will reset all parameters to the undefined value, thus avoiding unwanted copy.

EOCV.SelectSampleTexts

Toggles selection of the template texts whose bounding box intersects a given rectangle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SelectSampleTexts (
    int originX,
    int originY,
    int width,
    int height,
    Euresys.Open_eVision_2_10.ESelectionFlag textsSelectionFlag
)
```

Parameters

originX

Absolute coordinates of the selection rectangle (relative to the topmost parent image).

originY

Absolute coordinates of the selection rectangle (relative to the topmost parent image).

width

Limits of the selection rectangle.

height

Limits of the selection rectangle.

textsSelectionFlag

Selection mode for the template texts, as defined by [ESelectionFlag](#). By default, all texts are checked for intersection

Remarks

Unselected become selected and conversely. Reading or changing properties of the sample texts and the characters they contain can be done on basis of selection.

EOCV.SelectSampleTextsChars

Toggles selection of the template text characters whose bounding box intersects a given rectangle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SelectSampleTextsChars (
    int originX,
    int originY,
    int width,
    int height,
    Euresys.Open_eVision_2_10.ESelectionFlag textsSelectionFlag,
    Euresys.Open_eVision_2_10.ESelectionFlag charsSelectionFlag
)
```

Parameters

originX

Absolute coordinates of the selection rectangle (relative to the topmost parent image).

originY

Absolute coordinates of the selection rectangle (relative to the topmost parent image).

width

Limits of the selection rectangle.

height

Limits of the selection rectangle.

textsSelectionFlag

Selection mode for the sample texts, as defined by [ESelectionFlag](#). By default, all texts are checked for intersection

charsSelectionFlag

Selection mode for the sample text characters, as defined by [ESelectionFlag](#). By default, all characters of all texts are checked for intersection.

Remarks

Unselected become selected and conversely. Reading or changing properties of the sample texts and the characters they contain can be done on basis of selection.

EOCV.SelectTemplateChars

Toggles selection of the free characters whose bounding box intersects a given rectangle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SelectTemplateChars (
    int originX,
    int originY,
    int width,
    int height,
    Euresys.Open_eVision_2_10.ESelectionFlag charsSelectionFlag
)
```

Parameters

originX

Absolute coordinates of the selection rectangle (relative to the topmost parent image).

originY

Absolute coordinates of the selection rectangle (relative to the topmost parent image).

width

Limits of the selection rectangle.

height

Limits of the selection rectangle.

charsSelectionFlag

Selection mode for the free characters, as defined by [ESelectionFlag](#). By default, all free characters are processed.

Remarks

Unselected becomes selected and conversely. The free characters are sets of blobs taken from the free objects list. One step of the model definition is to specify which free objects will be handled by the OCV context and how they will be grouped to form characters. Grouping can be done automatically by using a overlapping criterion, or explicitly. When an object is added to a character, it is removed from the free objects list.

EOCV.SelectTemplateObjects

Toggles selection of the objects whose bounding box intersects a given rectangle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SelectTemplateObjects (
    int originX,
    int originY,
    int width,
    int height,
    Euresys.Open_eVision_2_10.ESelectionFlag objectsSelectionFlag
)
```

Parameters

originX

Absolute coordinates of the selection rectangle (relative to the topmost parent image).

originY

Absolute coordinates of the selection rectangle (relative to the topmost parent image).

width

Limits of the selection rectangle.

height

Limits of the selection rectangle.

objectsSelectionFlag

Selection mode for the objects, as defined by [ESelectionFlag](#). By default, all objects are processed.

Remarks

Unselected becomes selected and conversely. The free objects are blobs built using a separate coded image. One step of the model definition is to specify which objects will be handled by the OCV context.

EOCV.SelectTemplateTexts

Toggles selection of the template texts whose bounding box intersects a given rectangle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SelectTemplateTexts (
    int originX,
    int originY,
    int width,
    int height,
    Euresys.Open_eVision_2_10.ESelectionFlag textsSelectionFlag
)
```

Parameters

originX

Absolute coordinates of the selection rectangle (relative to the topmost parent image).

originY

Absolute coordinates of the selection rectangle (relative to the topmost parent image).

width

Limits of the selection rectangle.

height

Limits of the selection rectangle.

textsSelectionFlag

Selection mode for the template texts, as defined by [ESelectionFlag](#). By default, all texts are processed.

Remarks

Unselected becomes selected and conversely. The template texts are sets of characters taken from the free characters list. One step of the model definition is to specify which free characters will be handled by the OCV context and how they will be grouped to form texts. Grouping must be done explicitly. When a character is added to a text, it is removed from the free characters list.

EOCV.SetFreeChar

Sets an individual free character of the OCV context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetFreeChar(
    int index,
    Euresys.Open_eVision_2_10.EOCVChar freeChar
)
```

Parameters

index

Index of the individual free character.

freeChar

New free character.

Remarks

The free characters are sets of blobs taken from the free objects list. One step of the model definition is to specify which free objects will be handled by the OCV context, and how they will be grouped to form characters. Grouping can be done automatically by using a overlapping criterion, or explicitly. When an object is added to a character, it is removed from the free objects list.

EOCV.SetText

Modifies the individual text at the given index by the given value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetText(
    int index,
    Euresys.Open_eVision_2_10.EOCVText text
)
```

Parameters

index

Index, between **0** and [EOCV::NumTexts](#) (excluded) of the individual text to be modified.

text

The new text.

EOCV.SetTextCharParameters

Copies the desired parameters from the temporary character to the character of given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetTextCharParameters (
    Euresys.Open_eVision_2_10.EOCVChar ocvChar,
    uint textIndex,
    uint charIndex
)
```

Parameters

ocvChar

Temporary [EOCVChar](#) object.

textIndex

Character index in range **0** to **NumTexts-1**.

charIndex

Character index in range **0** to **NumTextChars-1**.

Remarks

Only the parameters for which the value is not undefined are copied. Before setting parameters, be sure to call the [EOCVChar::ResetParameters](#) member of the temporary [EOCVChar](#) object. This will reset all parameters to the undefined value, thus avoiding unwanted copy.

EOCV.SetTextParameters

Copies the desired parameters from the temporary text to the text of given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetTextParameters (
    Euresys.Open_eVision_2_10.EOCVText Text,
    uint textIndex
)
```

Parameters

Text

Text index in range **0** to **NumTexts**, excluded.

textIndex

Text index in range **0** to **NumTexts**, excluded.

Remarks

Only the parameters for which the value is not undefined value are copied. Before setting parameters, be sure to call the [EOCVText::ResetParameters](#) member of the temporary [EOCVText](#) object. This will reset all parameters to the undefined value, thus avoiding unwanted copy.

EOCV.StatisticsCount

Number of samples that have been taken into account in the statistics so far.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint StatisticsCount
{ get; }
```

Remarks

For averages to be computed, this member must amount to at least **1**. For standard deviations to be computed, this member must amount to at least **2**.

EOCV.TemplateBackground

Reference background gray levels determined during learning of the template.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float TemplateBackground
{ get; set; }
```

Remarks

Image contrast is an important factor during both learning and inspection. The background and foreground information must be separated using an appropriate threshold, possibly determined automatically. After a threshold is given, the average gray level of the background and foreground areas are computed separately. These serve as reference gray levels to measure the image contrast and normalize the gray level quality indicators. The background and foreground reference gray levels are computed for both the template and sample images. They are used for contrast assessment as well as gray-level normalization.

EOCV.TemplateContrast

Global contrast of the template image, or **FLOAT32_UNDEFINED** if it has not been computed.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float TemplateContrast
{ get; }
```

Remarks

The contrast is defined as the ratio of the reference gray-levels difference over their sum. This parameter is maximum (100 %) for a perfectly contrasted image (white on black).

EOCV.TemplateForeground

Reference foreground gray levels determined during learning of the template.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float TemplateForeground  
  
{ get; set; }
```

Remarks

Image contrast is an important factor during both learning and inspection. The background and foreground information must be separated using an appropriate threshold, possibly determined automatically. After a threshold is given, the average gray level of the background and foreground areas are computed separately. These serve as reference gray levels to measure the image contrast and normalize the gray level quality indicators. The background and foreground reference gray levels are computed for both the template and sample images. They are used for contrast assessment as well as gray-level normalization.

EOCV.TemplateImage

Source template image associated to the OCV context during model edition.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EROIBW8 TemplateImage  
  
{ get; set; }
```

Remarks

This property must be set as a first step before any model edition operation. During edition, the source image should not be altered. Before learning, the OCV context must have access to the template image to pre-process it. After learning, the OCV context keeps an internal copy for comparison purposes.

EOCV.TemplateThreshold

Threshold level applied to the template during learning.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EBWS TemplateThreshold  
  
{ get; set; }
```

Remarks

Image contrast is an important factor during both learning and inspection. The background and foreground information must be separated using an appropriate threshold, possibly determined automatically. After a threshold is given, the average gray level of the background and foreground areas are computed separately. These serve as reference gray levels to measure the image contrast and normalize the gray level quality indicators. The background and foreground reference gray levels are computed for both the template and sample images. They are used for contrast assessment as well as gray-level normalization.

EOCV.UpdateStatistics

Adds the parameters of the last inspection to the statistics.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```



```
void UpdateStatistics (
)
```

Remarks

This function must be called explicitly for the current sample to be taken into account.

EOCV.UsedQualityIndicators

Choice of quality indicators.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint UsedQualityIndicators
{ get; set; }
```

Remarks

For efficiency reasons, the quality indicators can be computed on demand only. This property is the logical combination (bitwise OR) of the values in [EQualityIndicator](#). When a bit is set in this mask, the corresponding feature is computed.

EOCV.WhiteOnBlack

Flag indicating whether the learning and inspection steps use white characters on a black background, or black characters on a white background.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
bool WhiteOnBlack
```

```
{ get; set; }
```

Remarks

If **TRUE** (default), the learning and inspection steps use white characters on a black background. If **FALSE**, black characters on a white background are taken into account. It is important to note that this parameter affects the learning *and* the inspection steps. Moreover, the use of the "black characters on a white background" mode requires the choice of the correct [ECodedImage](#) class.

4.134. EOCVChar Class

Holds all information related to a single character, such as nominal and average position, reference quality indicators,... During the learning phases, it also keeps the list of its constituent blobs.

Remarks

The statistics available for each character are the average, standard deviation (unbiased), minimum and maximum computed on the corresponding parameters for all images for which [EOCV::UpdateStatistics](#) has been invoked after inspection. The average is only available when at least one set of results has been accumulated. The standard deviation is only available when at least two sets of results have been accumulated.

Namespace: Euresys.Open_eVision_2_10

Properties

[BackgroundAreaAverage](#)

Background area average.

[BackgroundAreaDeviation](#)

Background area standard deviation.

[BackgroundAreaTolerance](#)

Maximum allowed difference between the sample and template background areas.

BackgroundSumAverage

Background sum average.

BackgroundSumDeviation

Background sum standard deviation.

BackgroundSumTolerance

Maximum allowed difference between the sample and template background sums.

Correlation

Normalized correlation involving the template and sample character images.

CorrelationAverage

Correlation average.

CorrelationDeviation

Correlation standard deviation.

CorrelationTolerance

Maximum allowed difference between the normalized correlation and unity.

Diagnostics

Logical combination (bitwise OR) of defects, as defined by [EDiagnostic](#).

ForegroundAreaAverage

Foreground area average.

ForegroundAreaDeviation

Foreground area standard deviation.

ForegroundAreaTolerance

Maximum allowed difference between the sample and template foreground areas.

ForegroundSumAverage	Foreground sum average.
ForegroundSumDeviation	Foreground sum standard deviation.
ForegroundSumTolerance	Maximum allowed difference between the sample and template foreground sums.
LocationScoreAverage	Location score average.
LocationScoreDeviation	Location score standard deviation.
LocationScoreTolerance	Allowed absolute difference between the template and sample scores.
MarginWidth	Width of the extra space to be used around the characters bounding box (on all four sides) when computing a quality indicator.
NumContourPoints	Number of contour points of this character used for location.
SampleBackgroundArea	Number of background pixels of this character corresponding to a background pixel of the template character.
SampleBackgroundSum	Sum of the normalized gray-level values of the pixels of this character corresponding to a background pixel of the template character.

SampleForegroundArea	Number of foreground pixels of this character corresponding to a foreground pixel of the template character.
SampleForegroundSum	Sum of the normalized gray-level values of the pixels of this character corresponding to a foreground pixel of the template character.
SampleLocationScore	Value of the location score measured on the sample during inspection.
Selected	Selection state: TRUE when selected.
ShiftX	Measured horizontal translation.
ShiftXAverage	Shift X average.
ShiftXBias	Horizontal translation bias.
ShiftXDeviation	Shift X standard deviation.
ShiftXMax	Maximum Shift X.
ShiftXMin	Minimum Shift X.

ShiftXStride

First pass stride for the horizontal translation.

ShiftXTolerance

Horizontal translation tolerance.

ShiftY

Measured vertical translation.

ShiftYAverage

Shift Y average.

ShiftYBias

Vertical translation bias. **0** corresponds to the nominal position.

ShiftYDeviation

Shift Y standard deviation.

ShiftYMax

Maximum Shift Y.

ShiftYMin

Minimum Shift Y.

ShiftYStride

First pass stride for the vertical translation.

ShiftYTolerance

Vertical translation tolerance.

TemplateBackgroundArea

Number of pixels in the background of this character.

TemplateBackgroundSum

Sum of the normalized gray-level values of the background pixels of this character.

TemplateForegroundArea

Number of pixels in the template foreground of this character.

TemplateForegroundSum

Sum of the normalized gray-level values of the foreground pixels of this character.

TemplateLocationScore

Value of the location score measured on the template during learning.

WhiteOnBlack

M Character contrast: **TRUE** for light characters on a dark background.

e

Methods

EOCVChar

Constructs an OCVChar context.

operator=

Copies all the data from another EOCVChar object into the current EOCVChar object

ResetParameters

Sets all character parameters to the undefined value for use before a [EOCV::SetTextCharParameters](#) or [EOCV::ScatterTextCharsParameters](#) operation.

EOCVChar.BackgroundAreaAverage

Background area average.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float BackgroundAreaAverage  
    { get; set; }
```

EOCVChar.BackgroundAreaDeviation

Background area standard deviation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float BackgroundAreaDeviation  
    { get; set; }
```

EOCVChar.BackgroundAreaTolerance

Maximum allowed difference between the sample and template background areas.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
```

```
uint BackgroundAreaTolerance
```

```
{ get; set; }
```

EOCVChar.BackgroundSumAverage

Background sum average.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float BackgroundSumAverage
```

```
{ get; set; }
```

EOCVChar.BackgroundSumDeviation

Background sum standard deviation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float BackgroundSumDeviation
```

```
{ get; set; }
```

EOCVChar.BackgroundSumTolerance

Maximum allowed difference between the sample and template background sums.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float BackgroundSumTolerance  
    { get; set; }
```

EOCVChar.Correlation

Normalized correlation involving the template and sample character images.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Correlation  
    { get; set; }
```

EOCVChar.CorrelationAverage

Correlation average.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float CorrelationAverage
```

```
{ get; set; }
```

EOCVChar.CorrelationDeviation

Correlation standard deviation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float CorrelationDeviation
```

```
{ get; set; }
```

EOCVChar.CorrelationTolerance

Maximum allowed difference between the normalized correlation and unity.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float CorrelationTolerance
```

```
{ get; set; }
```

EOCVChar.Diagnostics

Logical combination (bitwise OR) of defects, as defined by [EDiagnostic](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint Diagnostics
    { get; set; }
```

Remarks

After inspection, each character is tagged with a logical combination of diagnostics, each corresponding to a kind of defect.

EOCVChar.EOCVChar

Constructs an OCVChar context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EOCVChar (
)
void EOCVChar (
    Euresys.Open_eVision_2_10.EOCVChar other
)
```

Parameters

other

EOCVChar object to be copied.

Remarks

Default and copy constructors.

EOCVChar.ForegroundAreaAverage

Foreground area average.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ForegroundAreaAverage  
    { get; set; }
```

EOCVChar.ForegroundAreaDeviation

Foreground area standard deviation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ForegroundAreaDeviation  
    { get; set; }
```

EOCVChar.ForegroundAreaTolerance

Maximum allowed difference between the sample and template foreground areas.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint ForegroundAreaTolerance  
  
{ get; set; }
```

EOCVChar.ForegroundSumAverage

Foreground sum average.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ForegroundSumAverage  
  
{ get; set; }
```

EOCVChar.ForegroundSumDeviation

Foreground sum standard deviation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ForegroundSumDeviation  
  
{ get; set; }
```

EOCVChar.ForegroundSumTolerance

Maximum allowed difference between the sample and template foreground sums.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ForegroundSumTolerance  
    { get; set; }
```

EOCVChar.LocationScoreAverage

Location score average.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float LocationScoreAverage  
    { get; set; }
```

EOCVChar.LocationScoreDeviation

Location score standard deviation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float LocationScoreDeviation
{ get; set; }
```

EOCVChar.LocationScoreTolerance

Allowed absolute difference between the template and sample scores.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float LocationScoreTolerance
{ get; set; }
```

Remarks

When the sample value lies outside the acceptance interval, a character not found diagnostic is issued. During the location phase, a score is computed on the sample image. During learning, the same score is measured on the template image to serve as a reference. The closer the template and sample scores, the more successful the location. The location score is a first indication on the conformance of the inspected sample. In particular, a very small sample score may indicate that the character is absent.

EOCVChar.MarginWidth

Width of the extra space to be used around the characters bounding box (on all four sides) when computing a quality indicator.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
uint MarginWidth
{ get; set; }
```

EOCVChar.NumContourPoints

Number of contour points of this character used for location.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint NumContourPoints
{ get; set; }
```

Remarks

The location process relies on points from the external contours of the character constituent blobs. The number of points to be used can be adjusted. The smaller this value, the faster the location, but a too small value can cause mismatches.

EOCVChar.operator=

Copies all the data from another EOCVChar object into the current EOCVChar object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EOCVChar operator=(  
    Euresys.Open_eVision_2_10.EOCVChar other  
)
```

Parameters

other

EOCVChar object to be copied

EOCVChar.ResetParameters

Sets all character parameters to the undefined value for use before a [EOCV::SetTextCharParameters](#) or [EOCV::ScatterTextsCharsParameters](#) operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void ResetParameters (  
)
```

EOCVChar.SampleBackgroundArea

Number of background pixels of this character corresponding to a background pixel of the template character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
uint SampleBackgroundArea  
  
    { get; set; }
```

EOCVChar.SampleBackgroundSum

Sum of the normalized gray-level values of the pixels of this character corresponding to a background pixel of the template character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float SampleBackgroundSum  
    { get; set; }
```

EOCVChar.SampleForegroundArea

Number of foreground pixels of this character corresponding to a foreground pixel of the template character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint SampleForegroundArea  
    { get; set; }
```

EOCVChar.SampleForegroundSum

Sum of the normalized gray-level values of the pixels of this character corresponding to a foreground pixel of the template character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float SampleForegroundSum
{ get; set; }
```

EOCVChar.SampleLocationScore

Value of the location score measured on the sample during inspection.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float SampleLocationScore
{ get; set; }
```

Remarks

During the location phase, a score is computed on the sample image. During learning, the same score is measured on the template image to serve as a reference. The closer the template and sample scores, the more successful the location. The location score is a first indication on the conformance of the inspected sample. In particular, a very small sample score may indicate that the character is absent.

EOCVChar.Selected

Selection state: **TRUE** when selected.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

bool Selected

{ get; set; }

EOCVChar.ShiftX

Measured horizontal translation.

Namespace: Euresys.Open_eVision_2_10

[C#]

float ShiftX

{ get; set; }

EOCVChar.ShiftXAverage

Shift X average.

Namespace: Euresys.Open_eVision_2_10

[C#]

float ShiftXAverage

{ get; set; }

EOCVChar.ShiftXBias

Horizontal translation bias.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ShiftXBias  
  
{ get; set; }
```

Remarks

0 corresponds to the nominal position.

EOCVChar.ShiftXDeviation

Shift X standard deviation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ShiftXDeviation  
  
{ get; set; }
```

EOCVChar.ShiftXMax

Maximum Shift X.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ShiftXMax
```

```
{ get; set; }
```

EOCVChar.ShiftXMin

Minimum Shift X.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ShiftXMin  
    { get; set; }
```

EOCVChar.ShiftXStride

First pass stride for the horizontal translation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint ShiftXStride  
    { get; set; }
```

EOCVChar.ShiftXTolerance

Horizontal translation tolerance.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ShiftXTolerance
```

```
{ get; set; }
```

EOCVChar.ShiftY

Measured vertical translation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ShiftY
```

```
{ get; set; }
```

EOCVChar.ShiftYAverage

Shift Y average.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ShiftYAverage
```

```
{ get; set; }
```


EOCVChar.ShiftYBias

Vertical translation bias. **0** corresponds to the nominal position.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ShiftYBias  
    { get; set; }
```

EOCVChar.ShiftYDeviation

Shift Y standard deviation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ShiftYDeviation  
    { get; set; }
```

EOCVChar.ShiftYMax

Maximum Shift Y.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ShiftYMax
```

```
{ get; set; }
```

EOCVChar.ShiftYMin

Minimum Shift Y.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ShiftYMin
```

```
{ get; set; }
```

EOCVChar.ShiftYStride

First pass stride for the vertical translation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint ShiftYStride
```

```
{ get; set; }
```

EOCVChar.ShiftYTolerance

Vertical translation tolerance.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ShiftYTolerance  
    { get; set; }
```

EOCVChar.TemplateBackgroundArea

Number of pixels in the background of this character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint TemplateBackgroundArea  
    { get; set; }
```

EOCVChar.TemplateBackgroundSum

Sum of the normalized gray-level values of the background pixels of this character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float TemplateBackgroundSum  
    { get; set; }
```

EOCVChar.TemplateForegroundArea

Number of pixels in the template foreground of this character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint TemplateForegroundArea  
    { get; set; }
```

EOCVChar.TemplateForegroundSum

Sum of the normalized gray-level values of the foreground pixels of this character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float TemplateForegroundSum  
    { get; set; }
```

EOCVChar.TemplateLocationScore

Value of the location score measured on the template during learning.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float TemplateLocationScore  
  
    { get; set; }
```

Remarks

If necessary, this value may be set explicitly. During the location phase, a score is computed on the sample image. During learning, the same score is measured on the template image to serve as a reference. The closer the template and sample scores, the more successful the location. The location score is a first indication on the conformance of the inspected sample. In particular, a very small sample score may indicate that the character is absent.

EOCVChar.WhiteOnBlack

Character contrast: **TRUE** for light characters on a dark background.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool WhiteOnBlack  
  
    { get; set; }
```

4.135. EOCVText Class

Holds all information related to a single piece of text, such as nominal and average position, reference quality indicators,... It also keeps the list of its constituent characters.

Remarks

Each text can be translated horizontally and vertically, rotated, scaled horizontally and vertically and sheared with respect to its nominal position, to cope with mechanical displacement of the marking device. Location is performed in the range **Bias** +/- **Tolerance** around the nominal position. To speed up location, a two pass search may be performed, first with a large stride, then with a unit stride.

Note. The stride parameters apply to the two translation degrees of freedom only.

A set of parameters are computed on demand (see Inspection Options) during inspection. The values of these parameters are used to detect defects of various kinds by checking that they remain in a given tolerance interval. If not, a diagnostic is issued. Note that the quality indicators associated with the texts are the sums of the corresponding parameters for each of the constituent characters. The statistics available for each text are the average, standard deviation (unbiased), minimum and maximum computed on the corresponding parameters for all images for which [EOCV::UpdateStatistics](#) has been invoked after inspection. The average is only available when at least one set of results has been accumulated. The standard deviation is only available when at least two sets of results have been accumulated.

Namespace: Euresys.Open_eVision_2_10

Properties

[BackgroundAreaAverage](#)

Background area average.

[BackgroundAreaDeviation](#)

Background area standard deviation.

[BackgroundAreaTolerance](#)

Maximum allowed difference between the sample and template background areas.

[BackgroundSumAverage](#)

Background sum average.

BackgroundSumDeviation	Background sum standard deviation.
BackgroundSumTolerance	Maximum allowed difference between the sample and template background sums.
Correlation	Normalized correlation involving the template and sample images of the characters of this text.
CorrelationAverage	Correlation average.
CorrelationDeviation	Correlation standard deviation.
CorrelationTolerance	Maximum allowed difference between the normalized correlation and unity.
Diagnostics	Logical combination (bitwise OR) of defects, as defined by EDiagnostic .
ForegroundAreaAverage	Foreground area average.
ForegroundAreaDeviation	Foreground area standard deviation.
ForegroundAreaTolerance	Maximum allowed difference between the sample and template foreground areas.
ForegroundSumAverage	Foreground sum average.

ForegroundSumDeviation	Foreground sum standard deviation.
ForegroundSumTolerance	Maximum allowed difference between the sample and template foreground sums.
IsotropicScaling	Flag indicating whether the scaling degree of freedom should be considered isotropic (ScaleX and ScaleY identical) or not.
LocationScoreAverage	Location score average.
LocationScoreDeviation	Location score standard deviation.
LocationScoreTolerance	Allowed absolute difference between the template and sample scores.
MarginWidth	Unused.
NumContourPoints	Number of contour points used for location of this text.
SampleBackgroundArea	Number of background pixels of this text corresponding to a background pixel of the characters of the template text.
SampleBackgroundSum	Sum of the normalized gray-level values of the pixels of this text corresponding to a background pixel of the characters of the template text.

SampleForegroundArea

Number of foreground pixels of this text corresponding to a foreground pixel of the characters of the template text.

SampleForegroundSum

Sum of the normalized gray-level values of the pixels of this text corresponding to a foreground pixel of the characters of the template text.

SampleLocationScore

Value of the location score measured on the sample during inspection.

ScaleX

Measured horizontal scaling, expressed as a dimensionless ratio.

ScaleXAverage

Scale X average.

ScaleXBias

Horizontal scaling bias.

ScaleXCount

Number of values to be tried in the Bias +/- Tolerance range, bounds included.

ScaleXDeviation

Scale X standard deviation.

ScaleXMax

Maximum Y-scale.

ScaleXMin

Minimum X-scale.

ScaleXTolerance

Horizontal scaling tolerance.

ScaleY

Measured vertical scaling, expressed as a dimensionless ratio.

ScaleYAverage

Scale Y average.

ScaleYBias

Vertical scaling bias.

ScaleYCount

Number of values to be tried in the Bias +/- Tolerance range, bounds included.

ScaleYDeviation

Scale Y standard deviation.

ScaleYMax

Maximum Y-scale.

ScaleYMin

Minimum Y-scale.

ScaleYTolerance

Vertical scaling tolerance.

Selected

Selection state. **TRUE** when selected.

Shear

Measured shearing, clockwise from the vertical direction, expressed in the current angle unit.

ShearAverage

Shear average.

ShearBias

Shearing bias.

ShearCount

Number of values to be tried in the Bias +/- Tolerance range, bounds included.

ShearDeviation

Shear standard deviation.

ShearMax

Maximum shear.

ShearMin

Minimum shear.

ShearTolerance

Shearing tolerance.

ShiftX

Measured horizontal translation, in pixels.

ShiftXAverage

Shift X average.

ShiftXBias

Horizontal translation bias.

ShiftXDeviation

Shift X standard deviation.

ShiftXMax

Maximum Shift Y.

ShiftXMin	Minimum Shift X.
ShiftXStride	First pass stride for the horizontal translation.
ShiftXTolerance	Horizontal translation tolerance.
ShiftY	Measured vertical translation, in pixels.
ShiftYAverage	Shift Y average.
ShiftYBias	Vertical translation bias.
ShiftYDeviation	Shift Y standard deviation.
ShiftYMax	Maximum Shift Y.
ShiftYMin	Minimum Shift Y.
ShiftYStride	First pass stride for the vertical translation.
ShiftYTolerance	Vertical translation tolerance.
Skew	Measured rotation, clockwise from the horizontal direction, expressed in the current angle unit.

SkewAverage	Skew average.
SkewBias	Rotation bias.
SkewCount	Number of values to be tried in the Bias +/- Tolerance range, bounds included.
SkewDeviation	Skew standard deviation.
SkewMax	Maximum skew.
SkewMin	Minimum skew.
SkewTolerance	Rotation tolerance.
TemplateBackgroundArea	Number of pixels in the background of the characters of this text.
TemplateBackgroundSum	Sum of the normalized gray-level values of the background pixels of the characters of this text.
TemplateForegroundArea	Number of pixels in the foreground of all characters of this text.
TemplateForegroundSum	Sum of the normalized gray-level values of the foreground pixels of the characters of this text.

TemplateLocationScore

Methods

Value of the location score measured on the template during learning. If necessary, this value may be set explicitly.

EOCVText

Constructs an OCVText context.

operator=

Copies all the data from another EOCVText object into the current EOCVText object

ResetParameters

⌊ Sets all text parameters to the undefined value for use before a [EOCV::SetTextParameters](#) or [EOCV::ScatterTextsParameters](#) operation.



CVText.BackgroundAreaAverage

Background area average.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float BackgroundAreaAverage  
    { get; set; }
```

EOCVText.BackgroundAreaDeviation

Background area standard deviation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float BackgroundAreaDeviation  
    { get; set; }
```

EOCVText.BackgroundAreaTolerance

Maximum allowed difference between the sample and template background areas.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint BackgroundAreaTolerance  
    { get; set; }
```

EOCVText.BackgroundSumAverage

Background sum average.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float BackgroundSumAverage
```

```
{ get; set; }
```

EOCVText.BackgroundSumDeviation

Background sum standard deviation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float BackgroundSumDeviation
```

```
{ get; set; }
```

EOCVText.BackgroundSumTolerance

Maximum allowed difference between the sample and template background sums.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float BackgroundSumTolerance
```

```
{ get; set; }
```


EOCVText.Correlation

Normalized correlation involving the template and sample images of the characters of this text.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Correlation  
    { get; set; }
```

EOCVText.CorrelationAverage

Correlation average.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float CorrelationAverage  
    { get; set; }
```

EOCVText.CorrelationDeviation

Correlation standard deviation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float CorrelationDeviation
```

```
{ get; set; }
```

EOCVText.CorrelationTolerance

Maximum allowed difference between the normalized correlation and unity.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float CorrelationTolerance
```

```
{ get; set; }
```

EOCVText.Diagnostics

Logical combination (bitwise OR) of defects, as defined by [EDiagnostic](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint Diagnostics
```

```
{ get; set; }
```

Remarks

After inspection, each text is tagged with a logical combination of diagnostics, each corresponding to a kind of defect. The defects of the characters belonging to this texts are also added to this combination.

EOCVText.EOCVText

Constructs an OCVText context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EOCVText(
)
void EOCVText(
    Euresys.Open_eVision_2_10.EOCVText ocvText
)
```

Parameters

ocvText

EOCVText object to be copied.

Remarks

Default and copy constructors.

EOCVText.ForegroundAreaAverage

Foreground area average.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float ForegroundAreaAverage
    { get; set; }
```

EOCVText.ForegroundAreaDeviation

Foreground area standard deviation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ForegroundAreaDeviation  
    { get; set; }
```

EOCVText.ForegroundAreaTolerance

Maximum allowed difference between the sample and template foreground areas.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint ForegroundAreaTolerance  
    { get; set; }
```

EOCVText.ForegroundSumAverage

Foreground sum average.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ForegroundSumAverage
```

```
{ get; set; }
```

EOCVText.ForegroundSumDeviation

Foreground sum standard deviation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ForegroundSumDeviation
```

```
{ get; set; }
```

EOCVText.ForegroundSumTolerance

Maximum allowed difference between the sample and template foreground sums.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ForegroundSumTolerance
```

```
{ get; set; }
```

EOCVText.IsotropicScaling

Flag indicating whether the scaling degree of freedom should be considered isotropic (**ScaleX** and **ScaleY** identical) or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool IsotropicScaling  
  
    { get; set; }
```

Remarks

In most cases, isotropic scaling (default mode), is recommended. Isotropic scaling executes faster and is more realistic. In case of isotropic scaling search, the **ScaleY...** values are meaningless.

EOCVText.LocationScoreAverage

Location score average.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float LocationScoreAverage  
  
    { get; set; }
```

EOCVText.LocationScoreDeviation

Location score standard deviation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float LocationScoreDeviation  
    { get; set; }
```

EOCVText.LocationScoreTolerance

Allowed absolute difference between the template and sample scores.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float LocationScoreTolerance  
    { get; set; }
```

Remarks

When the sample value lies outside the acceptance interval, a text not found diagnostic is issued.

EOCVText.MarginWidth

Unused.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint MarginWidth
```

```
{ get; set; }
```

EOCVText.NumContourPoints

Number of contour points used for location of this text.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint NumContourPoints  
    { get; set; }
```

EOCVText.operator=

Copies all the data from another EOCVText object into the current EOCVText object

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EOCVText operator=(  
    Euresys.Open_eVision_2_10.EOCVText other  
    )
```

Parameters

other

EOCVText object to be copied

EOCVText.ResetParameters

Sets all text parameters to the undefined value for use before a [EOCV::SetTextParameters](#) or [EOCV::ScatterTextParameters](#) operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ResetParameters (
)
```

EOCVText.SampleBackgroundArea

Number of background pixels of this text corresponding to a background pixel of the characters of the template text.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint SampleBackgroundArea
{ get; set; }
```

EOCVText.SampleBackgroundSum

Sum of the normalized gray-level values of the pixels of this text corresponding to a background pixel of the characters of the template text.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float SampleBackgroundSum  
  
{ get; set; }
```

EOCVText.SampleForegroundArea

Number of foreground pixels of this text corresponding to a foreground pixel of the characters of the template text.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint SampleForegroundArea  
  
{ get; set; }
```

EOCVText.SampleForegroundSum

Sum of the normalized gray-level values of the pixels of this text corresponding to a foreground pixel of the characters of the template text.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float SampleForegroundSum  
  
{ get; set; }
```

EOCVText.SampleLocationScore

Value of the location score measured on the sample during inspection.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float SampleLocationScore  
    { get; set; }
```

EOCVText.ScaleX

Measured horizontal scaling, expressed as a dimensionless ratio.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ScaleX  
    { get; set; }
```

EOCVText.ScaleXAverage

Scale X average.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ScaleXAverage
```

```
{ get; set; }
```

EOCVText.ScaleXBias

Horizontal scaling bias.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ScaleXBias
```

```
{ get; set; }
```

Remarks

0 corresponds to the nominal, true scale factor.

EOCVText.ScaleXCount

Number of values to be tried in the Bias +/- Tolerance range, bounds included.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint ScaleXCount
```

```
{ get; set; }
```

EOCVText.ScaleXDeviation

Scale X standard deviation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ScaleXDeviation  
  
{ get; set; }
```

EOCVText.ScaleXMax

Maximum Y-scale.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ScaleXMax  
  
{ get; set; }
```

EOCVText.ScaleXMin

Minimum X-scale.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ScaleXMin
```

```
{ get; set; }
```

EOCVText.ScaleXTolerance

Horizontal scaling tolerance.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ScaleXTolerance
```

```
{ get; set; }
```

EOCVText.ScaleY

Measured vertical scaling, expressed as a dimensionless ratio.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ScaleY
```

```
{ get; set; }
```

EOCVText.ScaleYAverage

Scale Y average.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ScaleYAverage  
    { get; set; }
```

EOCVText.ScaleYBias

Vertical scaling bias.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ScaleYBias  
    { get; set; }
```

Remarks

0 corresponds to the nominal, true scale factor.

EOCVText.ScaleYCount

Number of values to be tried in the Bias +/- Tolerance range, bounds included.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint ScaleYCount  
    { get; set; }
```

EOCVText.ScaleYDeviation

Scale Y standard deviation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ScaleYDeviation  
    { get; set; }
```

EOCVText.ScaleYMax

Maximum Y-scale.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ScaleYMax  
    { get; set; }
```


EOCVText.ScaleYMin

Minimum Y-scale.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ScaleYMin  
{ get; set; }
```

EOCVText.ScaleYTolerance

Vertical scaling tolerance.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ScaleYTolerance  
{ get; set; }
```

EOCVText.Selected

Selection state. **TRUE** when selected.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
bool Selected
```

```
{ get; set; }
```

EOCVText.Shear

Measured shearing, clockwise from the vertical direction, expressed in the current angle unit.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float Shear
```

```
{ get; set; }
```

EOCVText.ShearAverage

Shear average.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ShearAverage
```

```
{ get; set; }
```

EOCVText.ShearBias

Shearing bias.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ShearBias  
  
{ get; set; }
```

Remarks

0 corresponds to the nominal, upright position.

EOCVText.ShearCount

Number of values to be tried in the Bias +/- Tolerance range, bounds included.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint ShearCount  
  
{ get; set; }
```

EOCVText.ShearDeviation

Shear standard deviation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ShearDeviation
```

```
{ get; set; }
```

EOCVText.ShearMax

Maximum shear.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ShearMax
```

```
{ get; set; }
```

EOCVText.ShearMin

Minimum shear.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ShearMin
```

```
{ get; set; }
```

EOCVText.ShearTolerance

Shearing tolerance.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ShearTolerance  
  
{ get; set; }
```

EOCVText.ShiftX

Measured horizontal translation, in pixels.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ShiftX  
  
{ get; set; }
```

EOCVText.ShiftXAverage

Shift X average.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ShiftXAverage
```

```
{ get; set; }
```

EOCVText.ShiftXBias

Horizontal translation bias.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ShiftXBias
```

```
{ get; set; }
```

Remarks

0 corresponds to the nominal position.

EOCVText.ShiftXDeviation

Shift X standard deviation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ShiftXDeviation
```

```
{ get; set; }
```

EOCVText.ShiftXMax

Maximum Shift Y.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ShiftXMax  
    { get; set; }
```

EOCVText.ShiftXMin

Minimum Shift X.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ShiftXMin  
    { get; set; }
```

EOCVText.ShiftXStride

First pass stride for the horizontal translation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint ShiftXStride  
{ get; set; }
```

EOCVText.ShiftXTolerance

Horizontal translation tolerance.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ShiftXTolerance  
{ get; set; }
```

EOCVText.ShiftY

Measured vertical translation, in pixels.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ShiftY  
{ get; set; }
```


EOCVText.ShiftYAverage

Shift Y average.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ShiftYAverage  
    { get; set; }
```

EOCVText.ShiftYBias

Vertical translation bias.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ShiftYBias  
    { get; set; }
```

Remarks

0 corresponds to the nominal position.

EOCVText.ShiftYDeviation

Shift Y standard deviation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ShiftYDeviation
```

```
{ get; set; }
```

EOCVText.ShiftYMax

Maximum Shift Y.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ShiftYMax
```

```
{ get; set; }
```

EOCVText.ShiftYMin

Minimum Shift Y.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ShiftYMin
```

```
{ get; set; }
```

EOCVText.ShiftYStride

First pass stride for the vertical translation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint ShiftYStride  
    { get; set; }
```

EOCVText.ShiftYTolerance

Vertical translation tolerance.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ShiftYTolerance  
    { get; set; }
```

EOCVText.Skew

Measured rotation, clockwise from the horizontal direction, expressed in the current angle unit.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Skew  
  
{ get; set; }
```

EOCVText.SkewAverage

Skew average.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float SkewAverage  
  
{ get; set; }
```

EOCVText.SkewBias

Rotation bias.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float SkewBias  
  
{ get; set; }
```

Remarks

0 corresponds to the nominal, horizontal position.

EOCVText.SkewCount

Number of values to be tried in the Bias +/- Tolerance range, bounds included.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint SkewCount  
    { get; set; }
```

EOCVText.SkewDeviation

Skew standard deviation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float SkewDeviation  
    { get; set; }
```

EOCVText.SkewMax

Maximum skew.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float SkewMax
```

```
{ get; set; }
```

EOCVText.SkewMin

Minimum skew.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float SkewMin
```

```
{ get; set; }
```

EOCVText.SkewTolerance

Rotation tolerance.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float SkewTolerance
```

```
{ get; set; }
```

EOCVText.TemplateBackgroundArea

Number of pixels in the background of the characters of this text.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint TemplateBackgroundArea  
    { get; set; }
```

EOCVText.TemplateBackgroundSum

Sum of the normalized gray-level values of the background pixels of the characters of this text.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float TemplateBackgroundSum  
    { get; set; }
```

EOCVText.TemplateForegroundArea

Number of pixels in the foreground of all characters of this text.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint TemplateForegroundArea
{ get; set; }
```

EOCVText.TemplateForegroundSum

Sum of the normalized gray-level values of the foreground pixels of the characters of this text.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float TemplateForegroundSum
{ get; set; }
```

EOCVText.TemplateLocationScore

Value of the location score measured on the template during learning. If necessary, this value may be set explicitly.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float TemplateLocationScore
{ get; set; }
```


4.136. EPathVector Class

Vector objects are used to store 1-dimensional data.

Remarks

Using vectors is very similar to using 1-dimensional arrays, except that the size can vary at runtime. Memory allocation is handled internally. * To create a vector, use its constructor. * To fill a vector with values, first empty it, using the [EPathVector](#) member, and then add elements one at a time at the tail by calling the [EPathVector::AddElement](#) member. * To access a vector element, either for reading or writing, use the [] operator. * To inquire for the current number of elements, use member [EPathVector](#).

Base Class: [EVector](#)

Namespace: Euresys.Open_eVision_2_10

Properties

Closed

-

RawDataPtr

M Pointer to the vector data.

e

Methods

AddElement

Appends (adds at the tail) an element to the vector.

Draw

Draws a plot of the vector element values.

DrawWithCurrentPen

Draws a plot of the vector element values.

EPathVector

Constructs a vector.

GetElement

Returns the vector element at the given index.

operator[]

Gives access to the vector element at the given index.

operator=

Copies all the data from another EPathVector object into the current EPathVector object

SetElement

Modifies the vector element at the given index by the given value.

P

athVector.AddElement

Appends (adds at the tail) an element to the vector.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddElement(
    Euresys.Open_eVision_2_10.EPath element
)
```

Parameters

element

The element to be added.

EPathVector.Closed

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool Closed  
    { get; set; }
```

EPathVector.Draw

Draws a plot of the vector element values.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Draw(  
    IntPtr graphicContext,  
    float zoomX,  
    float zoomY,  
    float originX,  
    float originY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ERGBColor color,  
    float zoomX,  
    float zoomY,  
    float originX,  
    float originY  
)  
  
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    float zoomX,  
    float zoomY,  
    float originX,  
    float originY  
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

zoomX

Zooming factor along the X axis (**1.0f** means no zoom).

zoomY

Zooming factor along the Y axis (**1.0f** means no zoom).

originX

Abscissa of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

originY

Ordinate of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

color

The color in which to draw the overlay.

Remarks

The vector draws line segment between the element coordinates. The drawing appears on the image itself.

EPathVector.DrawWithCurrentPen

Draws a plot of the vector element values.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawWithCurrentPen(
    IntPtr graphicContext,
    float zoomX,
    float zoomY,
    float originX,
    float originY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

zoomX

Zooming factor along the X axis (**1.0f** means no zoom).

zoomY

Zooming factor along the Y axis (**1.0f** means no zoom).

originX

Abscissa of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

originY

Ordinate of the upper left corner of the plot's bounding rectangle, in pixels. By default, the upper left corner of the window is used.

Remarks

The vector draws line segment between the element coordinates. The drawing appears on the image itself.

EPathVector.EPathVector

Constructs a vector.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EPathVector(
)
void EPathVector(
    uint maxNumberOfElements
)
void EPathVector(
    Euresys.Open_eVision_2_10.EPathVector other
)
```

Parameters

maxNumberOfElements

Optionally, memory can be pre-allocated to accommodate a given number of elements.

other

EPathVector object to be copied

EPathVector.GetElement

Returns the vector element at the given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EPath GetElement(  
    int index  
)
```

Parameters

index

Index, between **0** and [EPathVector](#) (excluded) of the element to be accessed.

Remarks

If the given index is outside the bounds of the vector, the error code [Parameter1OutOfRange](#) is set.

EPathVector.operator[]

Gives access to the vector element at the given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
ref Euresys.Open_eVision_2_10.EPath operator[] (  
    uint index  
)
```

Parameters

index

Index, between **0** and [EPathVector](#) (excluded) of the element to be accessed.

EPathVector.operator=

Copies all the data from another EPathVector object into the current EPathVector object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPathVector operator=(
    Euresys.Open_eVision_2_10.EPathVector other
)
```

Parameters

other

EPathVector object to be copied

EPathVector.RawDataPtr

Pointer to the vector data.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
IntPtr RawDataPtr
{ get; }
```

EPathVector.SetElement

Modifies the vector element at the given index by the given value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```



```
void SetElement(  
    int index,  
    Euresys.Open_eVision_2_10.EPath value  
)
```

Parameters

index

Index, between **0** and [EPathVector](#) (excluded), of the element to be modified.

value

The new value for the element.

Remarks

If the given index is outside the bounds of the vector, the error code [Parameter1OutOfRange](#) is set.

4.137. EPatternFinder Class

Manages a complete finding context in EasyFind.

Base Class: [EPointShape](#)

Namespace: Euresys.Open_eVision_2_10

Properties

[AngleBias](#)

Angle bias, expressed in the current angle unit.

[AngleSearchExtent](#)

The angular extension of the search neighborhood. See the [EPatternFinder::LocalSearchMode](#) property description for further details.

[AngleTolerance](#)

Angle tolerance, expressed in the current angle unit.

AutoTransitionThickness

Indicates whether the [EPatternFinder::TransitionThickness](#) property is automatically computed or not.

ContrastMode

Contrast of the instance, as defined in [EFindContrastMode](#).

FindExtension

Extension value, that is the pattern margin size (in pixels) that is allowed to go out of the search field.

ForcedThreshold

Forced threshold, between [0, 255].

Interpolate

Whether interpolation is performed when searching for a pattern occurrence.

LearningDone

Indicates whether a pattern has already been learnt.

LightBalance

Light balance, between **[-1.0, 1.0]**.

LocalSearchMode

Sets the local search mode.

MaxFeaturePoints

Maximum number of feature points at the fine stage.

MaxInstances

Maximum number of instances to be found.

MinFeaturePoints

Minimum number of feature points at the coarse stage.

MinScore	Minimum score of found instances, between [-1.0, 1.0] .
PatternType	Pattern type, as defined in EPatternType .
Pivot	Reference point in the model.
ReductionMode	The reduction mode that is to be used when learning the model (automatic or manual), as defined in EReductionMode .
ReductionStrength	The reduction strength that is to be used when learning the model, between 0 and 1 .
ScaleBias	Scale bias, expressed in units (not in percent).
ScaleSearchExtent	The scaling extension of the search neighborhood. See the EPatternFinder::LocalSearchMode property description for further details.
ScaleTolerance	Scale tolerance, expressed in units (not in percent).
ThinStructureMode	Mode for ThinStructure , as defined in EThinStructureMode .
TransitionThickness	Transition thickness, expressed in pixels.
Type	Shape type.

XSearchExtent

The X-axis extension of the search neighborhood. See the [EPatternFinder::LocalSearchMode](#) property description for further details.

YSearchExtent

M The Y-axis extension of the search neighborhood. See the [EPatternFinder::LocalSearchMode](#) property description for further details.

thods

CopyLearntPattern

Copies the learnt pattern in the supplied image. If no pattern has been learned, an exception with code [NoPatternLearnt](#) will be thrown.

DrawModel

Draws the model features with an overlay in image coordinates.

DrawModelWithCurrentPen

Draws the model features with an overlay in image coordinates.

EPatternFinder

Constructs a [EPatternFinder](#) context.

Find

Locates a set of possible occurrences of the learnt pattern in the supplied search field.

Learn

Learns a given pattern and stores it in the [EPatternFinder](#) object.

operator=

Copies all the data from another [EPatternFinder](#) object into the current [EPatternFinder](#) object

EPatternFinder.AngleBias

Angle bias, expressed in the current angle unit.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
float AngleBias  
  
{ get; set; }
```

Remarks

The **AngleBias** defines the angle offset between the model and the instances. Finding the pattern is performed in range **AngleBias** +/- [EPatternFinder::AngleTolerance](#). This range should not exceed a full turn. Default: **0.0**.

EPatternFinder.AngleSearchExtent

The angular extension of the search neighborhood. See the [EPatternFinder::LocalSearchMode](#) property description for further details.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
int AngleSearchExtent  
  
{ get; set; }
```

EPatternFinder.AngleTolerance

Angle tolerance, expressed in the current angle unit.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float AngleTolerance  
  
{ get; set; }
```

Remarks

The **AngleTolerance** defines the angle allowance of the instances around the [EPatternFinder::AngleBias](#). Finding the pattern is performed in range [EPatternFinder::AngleBias](#) +/- **AngleTolerance**. This range should not exceed a full turn. A **NULL** tolerance can be set, in which case the angle bias value is assumed. Default: **0.0**.

EPatternFinder.AutoTransitionThickness

Indicates whether the [EPatternFinder::TransitionThickness](#) property is automatically computed or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool AutoTransitionThickness  
  
{ get; set; }
```

Remarks

If set to **TRUE**, [EPatternFinder::TransitionThickness](#) is automatically computed during the [EPatternFinder::Learn](#) method call. This computed value will be used (by [EPatternFinder::Find](#)) until a new [EPatternFinder::Learn](#) is done. Even if the user explicitly sets the [EPatternFinder::TransitionThickness](#), it will have no effect, as [EPatternFinder::Find](#) will use the computed value. If set to **FALSE**, [EPatternFinder::TransitionThickness](#) is set by the user. It is never automatically changed by a [EPatternFinder::Learn](#) method call. Default: **TRUE**.

EPatternFinder.ContrastMode

Contrast of the instance, as defined in [EFindContrastMode](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EFindContrastMode ContrastMode  
  
{ get; set; }
```

Remarks

This is a [ConsistentEdges](#) pattern type property. It defines the contrast of regions. Contrast can be normal (as in the model), inverse (inverse contrast of the model), or any (same or inverse contrast of the model). Default: [Normal](#).

EPatternFinder.CopyLearntPattern

Copies the learnt pattern in the supplied image. If no pattern has been learned, an exception with code [NoPatternLearnt](#) will be thrown.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void CopyLearntPattern(  
    Euresys.Open_eVision_2_10.EImageBW8 image  
)
```

Parameters

image

Pointer to the image in which the learnt pattern will be returned.

EPatternFinder.DrawModel

Draws the model features with an overlay in image coordinates.

Namespace: Euresys.Open_eVision_2_10

```

[C#]
void DrawModel (
    IntPtr graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void DrawModel (
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void DrawModel (
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

```

Parameters

graphicContext

Handle to the device context of the destination windows.

zoomX

Horizontal zooming factor.

zoomY

Vertical zooming factor. If set to **0**, the horizontal zooming factor will be used for isotropic zooming.

panX

Horizontal panning offset.

panY

Vertical panning offset.

color

The color in which to draw the overlay.

EPatternFinder.DrawModelWithCurrentPen

Draws the model features with an overlay in image coordinates.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawModelWithCurrentPen(
    IntPtr graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

Parameters

graphicContext

Handle to the device context of the destination windows.

zoomX

Horizontal zooming factor.

zoomY

Vertical zooming factor. If set to **0**, the horizontal zooming factor will be used for isotropic zooming.

panX

Horizontal panning offset.

panY

Vertical panning offset.

EPatternFinder.EPatternFinder

Constructs a [EPatternFinder](#) context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EPatternFinder(
)

void EPatternFinder(
    Euresys.Open_eVision_2_10.EPatternFinder other
)
```

Parameters

other

Another EPatternFinder object to be copied in the new EPatternFinder object.

Remarks

All properties are initialized to their respective default values.

EPatternFinder.Find

Locates a set of possible occurrences of the learnt pattern in the supplied search field.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EFoundPattern[] Find(
    Euresys.Open_eVision_2_10.EROIBW8 source
)

Euresys.Open_eVision_2_10.EFoundPattern[] Find(
    Euresys.Open_eVision_2_10.EROIBW8 source,
    Euresys.Open_eVision_2_10.ERegion region
)
```

Parameters

source

Image or part of an image in which the learnt model has to be searched for.

region

Region into the ROI where the search is performed.

Remarks

This method will fail if no pattern has been learnt previously. The result is a vector of [EFoundPattern](#) objects.

EPatternFinder.FindExtension

Extension value, that is the pattern margin size (in pixels) that is allowed to go out of the search field.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int FindExtension
{ get; set; }
```

Remarks

When a non-**NULL** value is attributed to the extension, the detection of instances partially out of the ROI is allowed. The extension value defines how much the ROI is extended. Default: **0**.

EPatternFinder.ForcedThreshold

Forced threshold, between [0, 255].

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint ForcedThreshold
{ get; set; }
```

Remarks

This property fixes an absolute gray-level threshold to help the [EPatternFinder](#) in the extraction of regions in the [ContrastingRegions](#). Default: **0**, which means that the thresholding is computed automatically. Once this property has been changed, a new learning process has to be done, to take the new value into account. Note that this property will remain to its value even after a new learning process. An efficient way to see the effect of changing this property is to use the [EPatternFinder::DrawModel](#) method.

EPatternFinder.Interpolate

Whether interpolation is performed when searching for a pattern occurrence.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool Interpolate
{ get; set; }
```

Remarks

By default, matching is done with a one-pixel precision for all degrees of freedom (translation, rotation and scaling). You can use an additional interpolation process to achieve sub-pixel accuracy. This generally leads to an improvement of the sub-pixel accuracy by a factor larger than 10. This is possible only when the found instances match closely the model. A score higher than 0.99 indicates that the instances are a close match of the model. In other words, the instance is considered to be more accurate when the score is higher. The added computational cost is low. Default: **TRUE**.

EPatternFinder.Learn

Learns a given pattern and stores it in the [EPatternFinder](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Learn(
    Euresys.Open_eVision_2_10.EROIBW8 pattern,
    Euresys.Open_eVision_2_10.EROIBW8 dontCare
)

void Learn(
    Euresys.Open_eVision_2_10.EROIBW8 pattern,
    Euresys.Open_eVision_2_10.ERegion region
)
```

Parameters

pattern

Model to be learnt (ROI).

dontCare

"Don't care" area mask (ROI).

region

Region into the ROI where the learning is performed

Remarks

Learning another pattern erases the information stored for the first one. A "don't care area" can be set as argument, allowing to mask and not take into account certain parts of the pattern while learning. The "don't care area" mask should have the same size as the model. The mask should be a bilevel image with pixel - values of '0' for ignored areas and '255' otherwise.

EPatternFinder.LearningDone

Indicates whether a pattern has already been learnt.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool LearningDone
{ get; }
```

EPatternFinder.LightBalance

Light balance, between **[-1.0, 1.0]**.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
float LightBalance  
  
{ get; set; }
```

Remarks

Consistent Edges and Thin StructuresIn the [ConsistentEdges](#) and [ThinStructure](#) modes, the **LightBalance** property governs the selection of the feature points while learning the model. It drives which edge points are eligible as feature points in the model, by defining a criterion for ignoring those edge points that are not sharp enough. As a consequence, this property will influence the spatial distribution of the feature points. In the aforementioned operating modes, the feature points are the places in the image that exhibit a strong variation in the gray level signal. Mathematically, these places are those at which the magnitude of the gradient is significant. The **LightBalance** property defines the way the latter threshold on the magnitude of the gradient is computed, through a careful analysis of the dynamics of gradient. The more the **LightBalance** tends to -1, the more tolerant will be the threshold, and the more edge points will be considered as candidates for becoming feature points. Conversely, as the **LightBalance** property becomes close to 1, only the points with a high gradient magnitude are taken into consideration. In other words, a small **LightBalance** defines a loose criterion for defining what an edge point is, whereas a great value implies a conservative criterion. By default, this property is fixed to **0.0**. This is an appropriate value for most images which are encountered in industrial machine vision. Contrasting RegionsWhen using the [ContrastingRegions](#) pattern type, this property allows the user to compensate for poor lighting conditions of the model. The more the **LightBalance** tends to 1, the lower the threshold will be on the model, and the more dark regions will be considered. Conversely, as the **LightBalance** parameter becomes close to -1, only the bright regions are taken into consideration. The **LightBalance** is automatically set to **0.0** after a learning process. Once the **LightBalance** is changed, a new learning process has to be done to take the new value into account. An efficient way to see the effect of changing this property is to use the [EPatternFinder::DrawModel](#) method.

EPatternFinder.LocalSearchMode

Sets the local search mode.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.ELocalSearchMode LocalSearchMode  
  
{ get; set; }
```

Remarks

In the multi-stage approach of EasyFind, pattern occurrence candidates are at first found at the coarsest stage. Then, at each of the following stages, their position and score are refined until the last and finest one. This refining is achieved by searching for better candidates in the neighborhood of each of the ones found in the previous stage. The local search mode allows the user to set the extent of this neighborhood. By default, the local search mode is set to [Basic](#).

EPatternFinder.MaxFeaturePoints

Maximum number of feature points at the fine stage.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint MaxFeaturePoints  
  
{ get; set; }
```

Remarks

Default: **1024**. Reserved use.

EPatternFinder.MaxInstances

Maximum number of instances to be found.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint MaxInstances
{ get; set; }
```

Remarks

Default: **1**.

EPatternFinder.MinFeaturePoints

Minimum number of feature points at the coarse stage.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint MinFeaturePoints
{ get; set; }
```

Remarks

Default: **8**. Reserved use.

EPatternFinder.MinScore

Minimum score of found instances, between **[-1.0, 1.0]**.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```



```
float MinScore
```

```
{ get; set; }
```

Remarks

Instances with a score under the **MinScore** will not be returned.

EPatternFinder.operator=

Copies all the data from another EPatternFinder object into the current EPatternFinder object

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPatternFinder operator=(  
    Euresys.Open_eVision_2_10.EPatternFinder other  
)
```

Parameters

other

EPatternFinder object to be copied

EPatternFinder.PatternType

Pattern type, as defined in [EPatternType](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EPatternType PatternType
```

```
{ get; set; }
```

Remarks

This property informs the [EPatternFinder](#) of the general nature of the model to be learnt. Default: [ConsistentEdges](#).

EPatternFinder.Pivot

Reference point in the model.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EPoint Pivot
```

```
{ get; set; }
```

Remarks

The coordinates of the reference point are relative to the upper left corner of the model. The location of an instance (Coordinates (X,Y)) is the location of its reference point defined in the model. By default, the pivot is a [EPoint](#) set to the pattern center. [EPoint](#) is a structure that contains two x and y float values.

EPatternFinder.ReductionMode

The reduction mode that is to be used when learning the model (automatic or manual), as defined in [EReductionMode](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EReductionMode ReductionMode  
  
{ get; set; }
```

Remarks

Specifies whether the best-guess method should be used to assert the level of reduction that will be used when learning the model. If this property is set to [Manual](#), it is up to the user to provide a suitable reduction strength. This value is only used when learning the model. Default: [Auto](#), which means that the best-guess algorithm is used by default.

EPatternFinder.ReductionStrength

The reduction strength that is to be used when learning the model, between **0** and **1**.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ReductionStrength  
  
{ get; set; }
```

Remarks

Specifies the reduction strength for learning the model (encoded as a percentage). Its precise semantics depends on the reduction mode (see the [EPatternFinder::ReductionMode](#) property): * In the automatic reduction mode, its value is undefined until a model is learned. When a model is learned (i.e. after a call to [EPatternFinder::Learn](#)), the value of this property can be read, in which case it reflects the reduction strength that has been automatically chosen by the best-guess algorithm. * In the manual reduction mode, this property must be set by the user and is kept constant throughout the entire lifetime of the object. The new value of the property is only used at the following call to [EPatternFinder::Learn](#). This value only has an effect when learning the model. Default: The default value depends on the value of the [EPatternFinder::ReductionMode](#) property. Allowed values: Floating-point number in the interval **[0..1]**.

EPatternFinder.ScaleBias

Scale bias, expressed in units (not in percent).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ScaleBias  
  
{ get; set; }
```

Remarks

The **ScaleBias** defines the scale factor between the model and the instances. Finding the pattern is performed in range **ScaleBias** +/- **ScaleTolerance**. This range should not exceed **[0.5..2.5]** (50 % to 250 % scaling). Default: **1.0** (100 %).

EPatternFinder.ScaleSearchExtent

The scaling extension of the search neighborhood. See the [EPatternFinder::LocalSearchMode](#) property description for further details.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int ScaleSearchExtent  
  
{ get; set; }
```

EPatternFinder.ScaleTolerance

Scale tolerance, expressed in units (not in percent).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ScaleTolerance  
  
{ get; set; }
```

Remarks

The **ScaleTolerance** defines the scale allowance of the instances around the [EPatternFinder::ScaleBias](#). Finding the pattern is performed in range [EPatternFinder::ScaleBias](#) +/- **ScaleTolerance**. This range should not exceed **[0.5..2]** (50 % to 200 % scaling). A **NULL** tolerance can be set, in which case the scale bias value is assumed. Default: **0.0**.

EPatternFinder.ThinStructureMode

Mode for [ThinStructure](#), as defined in [EThinStructureMode](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EThinStructureMode ThinStructureMode  
  
{ get; set; }
```

Remarks

[EThinStructureMode](#) informs EasyFind if thin elements in the model are darker or brighter than regions. Default: [Auto](#), which detects the best mode between dark or bright.

EPatternFinder.TransitionThickness

Transition thickness, expressed in pixels.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint TransitionThickness
    { get; set; }
```

Remarks

This property defines the tolerance on the location of transitions between regions in [ContrastingRegions](#). An efficient way to see the effect of changing this property is to use the [EPatternFinder::DrawModel](#) method. The **TransitionThickness** value, used by [EPatternFinder::Find](#), is either the automatically computed value (by a [EPatternFinder::Learn](#) method call) if [EPatternFinder::AutoTransitionThickness](#) is **TRUE**, or the user-defined value if [EPatternFinder::AutoTransitionThickness](#) is **FALSE**.

Note. If [EPatternFinder::AutoTransitionThickness](#) is **TRUE** and no [EPatternFinder::Learn](#) has been executed, the **TransitionThickness** value that [EPatternFinder::Find](#) will use is undefined, even if it has been manually set in the meantime.

EPatternFinder.Type

Shape type.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
override Euresys.Open_eVision_2_10.EShapeType Type
    { get; }
```

EPatternFinder.XSearchExtent

The X-axis extension of the search neighborhood. See the [EPatternFinder::LocalSearchMode](#) property description for further details.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
int XSearchExtent  
{ get; set; }
```

EPatternFinder.YSearchExtent

The Y-axis extension of the search neighborhood. See the [EPatternFinder::LocalSearchMode](#) property description for further details.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
int YSearchExtent  
{ get; set; }
```

4.138. EPeakVector Class

Represents a vector of profile peaks.

Base Class: [EVector](#)

Namespace: Euresys.Open_eVision_2_10

Properties

[RawDataPtr](#)

Pointer to the vector data.

Methods

AddElement

Appends (adds at the tail) an element to the vector.

EPeakVector

Constructs a vector.

GetElement

Returns the vector element at the given index.

operator[]

Gives access to the vector element at the given index.

operator=

Copies all the data from another EPeakVector object into the current EPeakVector object

SetElement

Modifies the vector element at the given index by the given value.

P

eakVector.AddElement

Appends (adds at the tail) an element to the vector.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddElement(
    Euresys.Open_eVision_2_10.EPeak element
)
```


Parameters

element

The element to be added.

EPeakVector.EPeakVector

Constructs a vector.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EPeakVector (
)
void EPeakVector (
    Euresys.Open_eVision_2_10.EPeakVector other
)
void EPeakVector (
    uint maxNumberOfElements
)
```

Parameters

other

EPeakVector object to be copied

maxNumberOfElements

Optionally, memory can be pre-allocated to accommodate a given number of elements.

EPeakVector.GetElement

Returns the vector element at the given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPeak GetElement(
    int index
)
```

Parameters

index

Index, between **0** and [EPeakVector](#) (excluded) of the element to be accessed.

Remarks

If the given index is outside the bounds of the vector, the error code [Parameter1OutOfRange](#) is set.

EPeakVector.operator[]

Gives access to the vector element at the given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
ref Euresys.Open_eVision_2_10.EPeak operator[] (
    uint index
)
```

Parameters

index

Index, between **0** and [EPeakVector](#) (excluded) of the element to be accessed.

EPeakVector.operator=

Copies all the data from another EPeakVector object into the current EPeakVector object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPeakVector operator=(
    Euresys.Open_eVision_2_10.EPeakVector other
)
```

Parameters

other

EPeakVector object to be copied

EPeakVector.RawDataPtr

Pointer to the vector data.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
IntPtr RawDataPtr
{ get; }
```

EPeakVector.SetElement

Modifies the vector element at the given index by the given value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void SetElement(  
    int index,  
    Euresys.Open_eVision_2_10.EPeak value  
)
```

Parameters

index

Index, between **0** and [EPeakVector](#) (excluded), of the element to be modified.

value

The new value for the element.

Remarks

If the given index is outside the bounds of the vector, the error code [Parameter1OutOfRange](#) is set.

4.139. EPlaneCropper Class

A [EPlaneCropper](#) object is used to crop some points of a [EPointCloud](#) object.

The points to keep are selected according to their positions with respect to a reference plane.

A [EPlaneCropper](#) object is characterized by its reference plane and is used to produce an output [EPointCloud](#) object from an input [EPointCloud](#) object.

The produced point cloud contains a subset of the input point cloud.

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

Plane

M Sets/gets the new reference [E3DPlane](#) of the [EPlaneCropper](#) object.

e

Methods

Crop

Crops an [EPointCloud](#). An output point cloud is produced from an input point cloud by only keeping the points that satisfy the specified condition.

The condition tests the (signed) distance of the points with respect to the reference plane of the cropper.

[EPlaneCropper](#)

Creates an [EPlaneCropper](#) object using a horizontal plane as a default reference.

[Load](#)

Loads the [EPlaneCropper](#) configuration. The given [ESerializer](#) must have been created for reading.

[operator=](#)

Assignment operator.

[Save](#)

[E](#) Saves the [EPlaneCropper](#) configuration. The given [ESerializer](#) must have been created for writing.

P

laneCropper.Crop

Crops an [EPointCloud](#). An output point cloud is produced from an input point cloud by only keeping the points that satisfy the specified condition.

The condition tests the (signed) distance of the points with respect to the reference plane of the cropper.

Namespace: Euresys.Open_eVision_2_10.Easy3D

[C#]

```
void Crop(  
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud cloudIn,  
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud cloudOut,  
    Euresys.Open_eVision_2_10.Easy3D.EPlaneCropperType type,  
    float maxDistance  
)
```

Parameters

cloudIn

The input point cloud.

cloudOut

The output point cloud.

type

An enum of type [EPlaneCropperType](#) that specifies which points of the input point cloud will be copied to the output point cloud.

maxDistance

Specifies the distance from the plane for the types "EPlaneCropperType_KeepClose" and "EPlaneCropperType_KeepFar".

It should be 0 for the types "EPlaneCropperType_KeepAbove" and "EPlaneCropperType_KeepBelow".

Remarks

The input point cloud 'cloudIn' should be different from the output point cloud 'cloudOut'. If not an exception will be thrown.

EPlaneCropper.EPlaneCropper

Creates an [EPlaneCropper](#) object using a horizontal plane as a default reference.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void EPlaneCropper(  
)
```

```
void EPlaneCropper(  
    Euresys.Open_eVision_2_10.Easy3D.E3DPlane plane  
)  
  
void EPlaneCropper(  
    Euresys.Open_eVision_2_10.Easy3D.EPlaneCropper other  
)
```

Parameters

plane

Reference [E3DPlane](#) used for the initialization.

other

Reference [EPlaneCropper](#) used for the initialization.

EPlaneCropper.Load

Loads the [EPlaneCropper](#) configuration. The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
void Load(  
    Euresys.Open_eVision_2_10.ESerializer serializer  
)
```

Parameters

serializer

The [ESerializer](#) object that is read from

EPlaneCropper.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EPlaneCropper operator=(
    Euresys.Open_eVision_2_10.Easy3D.EPlaneCropper other
)
```

Parameters

other

An other [EPlaneCropper](#).

EPlaneCropper.Plane

Sets/gets the new reference [E3DPlane](#) of the [EPlaneCropper](#) object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DPlane Plane
    { get; set; }
```

EPlaneCropper.Save

Saves the [EPlaneCropper](#) configuration. The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10.Easy3D


```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is written to

4.140. EPlaneFinder Class

A [EPlaneFinder](#) object is used to search a [E3DPlane](#) in a [EPointCloud](#).

The algorithm searches **the largest plane** in terms of number of "inliers". A point is an "inlier" when its distance to the plane is smaller than a specified threshold (parameter 'maximum distance').

Another parameter specifies the expected ratio of inliers over the total number of points in the point cloud (by default, this is set to **0.3**).

Reducing the value of this parameter makes the search more robust but will potentially consume more time. A decimation is applied by default to accelerate the search.

Furthermore the expected normal to the plane may be specified.

The method [EPlaneFinder::Find](#) processes a [EPointCloud](#) object and returns a [E3DPlane](#) object when a plane is found in the input point cloud.

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

[ExpectedCloudInliersRatio](#)

Sets/gets the expected ratio of inliers in the [EPointCloud](#).

[MaxDeviation](#)

Sets/gets the maximum distance of a inlier to the plane.

NormalTolerance

M Returns the angle tolerance around the expected normal (that has been set by [EPlaneFinder::SetNormal](#))

e

thods

DisableDecimator

Disables the default decimation.
The decimation should be disabled when the input point cloud is already decimated.

EnableDecimator

Enables the default decimation which reduces the input point cloud by drawing 10000 points randomly.
This decimation is enabled by default. The decimation accelerates the search.

EPlaneFinder

Creates an [EPlaneFinder](#) object.

Find

Searches the biggest plane in the supplied [EPointCloud](#). If the plane is found, a [E3DPlane](#) object is returned.
Otherwise an exception is thrown.

GetNormal

Returns the expected normal direction (that has been set by [EPlaneFinder::SetNormal](#))

IsDecimatorEnabled

Returns true if the default decimator is enabled (enabled by default).

IsNormalSet

Returns true if the expected normal direction has been set (not set by default).

Load	Loads the plane finder configuration. The given ESerializer must have been created for reading.
operator=	Assignment operator.
Save	Saves the plane finder configuration. The given ESerializer must have been created for writing.
SetNormal	Sets the expected normal direction and the tolerance around it. These values are used to limit the scope of the plane search. If the angular tolerance is not specified, a default value of 5 degrees is assumed. If the tolerance is specified, the value should be strictly positive.
UnsetNormal	Unsets the normal vector definition.

P

laneFinder.DisableDecimator

Disables the default decimation.
The decimation should be disabled when the input point cloud is already decimated.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void DisableDecimator(
)
```

EPlaneFinder.EnableDecimator

Enables the default decimation which reduces the input point cloud by drawing 10000 points randomly. This decimation is enabled by default. The decimation accelerates the search.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void EnableDecimator(
)
```

EPlaneFinder.EPlaneFinder

Creates an [EPlaneFinder](#) object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void EPlaneFinder(
)
void EPlaneFinder(
    float maxDeviation,
    float pcExpectedInCloud
)
void EPlaneFinder(
    Euresys.Open_eVision_2_10.Easy3D.EPlaneFinder other
)
```

Parameters

maxDeviation

Maximum distance of a inlier to the plane. This value has to be strictly positive.

pcExpectedInCloud

This is an estimation of the ratio of inliers in the point cloud.

This optional parameter has a default value of **0.3**.

The expected ratio of inliers has an influence on the duration and on the robustness of the search: a smaller value increases the chance of finding a smaller plane while a larger value is faster.

other

The [EPlaneFinder](#) object that should be copied.

EPlaneFinder.ExpectedCloudInliersRatio

Sets/gets the expected ratio of inliers in the [EPointCloud](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float ExpectedCloudInliersRatio  
  
{ get; set; }
```

Remarks

It is an estimation of the ratio of inliers in the point cloud.

The value set by default is **0.3**. The expected ratio of inliers has an influence on the duration and on the robustness of the search:

a smaller value increases the chance of finding the plane while a larger value makes it faster.

EPlaneFinder.Find

Searches the biggest plane in the supplied [EPointCloud](#). If the plane is found, a [E3DPlane](#) object is returned. Otherwise an exception is thrown.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DPlane Find(
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud pointCloud
)

Euresys.Open_eVision_2_10.Easy3D.E3DPlane Find(
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud pointCloud,
    ref float effectiveInliersRatio
)
```

Parameters

pointCloud

The input point cloud in which the plane should be searched (need at least 3 points)

effectiveInliersRatio

This passed by reference float will contain the effective ratio of inliers

EPlaneFinder.GetNormal

Returns the expected normal direction (that has been set by [EPlaneFinder::SetNormal](#))

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DPoint GetNormal(
)
```

EPlaneFinder.IsDecimatorEnabled

Returns true if the default decimator is enabled (enabled by default).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool IsDecimatorEnabled(
)
```

EPlaneFinder.IsNormalSet

Returns true if the expected normal direction has been set (not set by default).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool IsNormalSet(
)
```

EPlaneFinder.Load

Loads the plane finder configuration. The given ESerializer must have been created for reading.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Load(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is read from.

EPlaneFinder.MaxDeviation

Sets/gets the maximum distance of a inlier to the plane.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float MaxDeviation  
  
{ get; set; }
```

EPlaneFinder.NormalTolerance

Returns the angle tolerance around the expected normal (that has been set by [EPlaneFinder::SetNormal](#))

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float NormalTolerance  
  
{ get; }
```

EPlaneFinder.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D


```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EPlaneFinder operator=(
    Euresys.Open_eVision_2_10.Easy3D.EPlaneFinder other
)
```

Parameters

other

The [EPlaneFinder](#) object that should be copied.

EPlaneFinder.Save

Saves the plane finder configuration. The given ESerializer must have been created for writing.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is written to.

EPlaneFinder.SetNormal

Sets the expected normal direction and the tolerance around it. These values are used to limit the scope of the plane search.

If the angular tolerance is not specified, a default value of **5 degrees** is assumed.

If the tolerance is specified, the value should be strictly positive.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetNormal(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint normal,
    float angleTolerance
)

void SetNormal(
    float nx,
    float ny,
    float nz,
    float angleTolerance
)

void SetNormal(
    Euresys.Open_eVision_2_10.Easy3D.E3DPlane referencePlane,
    float angleTolerance
)
```

Parameters

normal

The normal vector specifies the expected perpendicular direction of the plane.

angleTolerance

The angle tolerance is the maximum angular deviation around the expected normal (strictly positive value). It's set to 5 degrees by default.

nx

The x component of the normal vector.

ny

The y component of the normal vector.

nz

The z component of the normal vector.

referencePlane

The reference plane specifies the expected perpendicular direction.

EPlaneFinder.UnsetNormal

Unsets the normal vector definition.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void UnsetNormal (
)
```

4.141. EPlaneFitter Class

A [EPlaneFitter](#) object is used to fit an [E3DPlane](#) on an [EPointCloud](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

[MinSampleCount](#)

M Sets/Gets the minimum number of samples required for fitting on each side of the shape.
e By default, a value of **3** is assumed.

Methods

[EPlaneFitter](#)

Constructor of an [EPlaneFitter](#) object.

[Fit](#)

Fits an [E3DPlane](#) on a given [EPointCloud](#).

[Load](#)

Loads the [EPlaneFitter](#) configuration. The given [ESerializer](#) must have been created for reading.

[operator=](#)

Assignment operator.

Save

E Saves the [EPlaneFitter](#) configuration. The given [ESerializer](#) must have been created for writing.
P

laneFitter.EPlaneFitter

Constructor of an [EPlaneFitter](#) object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void EPlaneFitter(  
    )  
void EPlaneFitter(  
    Euresys.Open_eVision_2_10.Easy3D.EPlaneFitter other  
    )
```

Parameters

other

Reference to the [EPlaneFitter](#) used for the initialization.

EPlaneFitter.Fit

Fits an [E3DPlane](#) on a given [EPointCloud](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
Euresys.Open_eVision_2_10.Easy3D.E3DPlane Fit(  
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud pc  
)  
  
Euresys.Open_eVision_2_10.Easy3D.E3DPlane Fit(  
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud pc,  
    out float averageDistance  
)
```

Parameters

pc

The reference to the point cloud.

averageDistance

The reference to a float which will store the average distance from this plane to the points that were used for the fit.

EPlaneFitter.Load

Loads the [EPlaneFitter](#) configuration. The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
void Load(  
    Euresys.Open_eVision_2_10.ESerializer serializer  
)
```

Parameters

serializer

The [ESerializer](#) object that is read from

EPlaneFitter.MinSampleCount

Sets/Gets the minimum number of samples required for fitting on each side of the shape.
By default, a value of **3** is assumed.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
int MinSampleCount  
  
    { get; set; }
```

EPlaneFitter.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.Easy3D.EPlaneFitter operator=(  
    Euresys.Open_eVision_2_10.Easy3D.EPlaneFitter other  
)
```

Parameters

other

The [EPlaneFitter](#) object that should be copied.

EPlaneFitter.Save

Saves the [EPlaneFitter](#) configuration. The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is written to

4.142. EPoint Class

An exact (floating-point) location in the 2D space.

Derived Class(es): [EFrame](#)

Namespace: Euresys.Open_eVision_2_10

Properties

[Center](#)

Center coordinates of a [EPoint](#) object.

[X](#)

Abscissa (X coordinate) of the [EPoint](#) object

Y

M Ordinate (Y coordinate) of the [EPoint](#) object

e

thods

Area

Compute the oriented area of the parallelogram built on two [EPoint](#).

Argument

Compute the polar argument of a [EPoint](#) object.

CopyTo

Copies all the data of the current [EPoint](#) object into another [EPoint](#) object and returns it.

Distance

Returns the distance between the addressed point and an [EPoint](#) object.

Dot

Compute the dot product of two [EPoint](#) object.

EPoint

Constructs a [EPoint](#) object.

MidPoint

Returns the middle coordinate between this [EPoint](#) object and another [EPoint](#) object.

Modulus

Compute the euclidian modulus of a [EPoint](#).

operator-

Subtracts from the current [EPoint](#) center coordinates the center coordinates of another [EPoint](#) object.

operator!=

Compares the current [EPoint](#) center coordinates with the center coordinates of another [EPoint](#) object.

operator*

Multiplies the current [EPoint](#) center coordinates by a given multiplier.

operator/

Divides the current [EPoint](#) center coordinates by a given divisor.

operator+

Adds to the current [EPoint](#) center coordinates the center coordinates of another [EPoint](#) object.

operator=

Copies all the data from another [EPoint](#) object into the current [EPoint](#) object.

operator==

Compares the current [EPoint](#) center coordinates with the center coordinates of another [EPoint](#) object.

Project

Compute the orthogonal projection of a [EPoint](#) on another shape.

Rotate

Returns another [EPoint](#) object containing the coordinated of the rotated point.

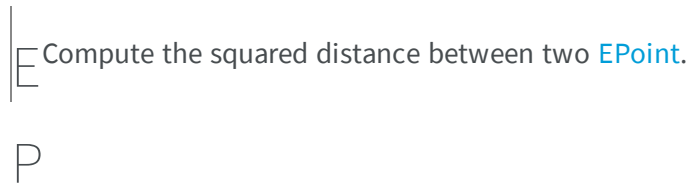
SetCenterXY

Sets the center coordinates of a [EPoint](#) object.

Square

Compute the sum of the squared coordinates of a [EPoint](#).

SquaredDistance



oint.Area

Compute the oriented area of the parallelogram built on two [EPoint](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float Area (
    Euresys.Open_eVision_2_10.EPoint Point
)
```

Parameters

Point

Second edge of the parallelogram.

Remarks

Compute the oriented area of the parallelogram built on two [EPoint](#). The area is counted as positive if the oriented vector pair ('first edge', 'second edge') is in the same sense that the axis frame. This oriented area can also be viewed as the z-coordinate of a vector product of two 3D vectors obtained in supplementing each edges with a third z-coordinate (setted to zero).

EPoint.Argument

Compute the polar argument of a [EPoint](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Argument(  
)
```

Remarks

Compute the angle (in radians) between the oriented X-axis and the vector going from the axis origin and the [EPoint](#). If the axis frame is orthogonal, this number is also the polar argument of the [EPoint](#).

EPoint.Center

Center coordinates of a [EPoint](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
virtual Euresys.Open_eVision_2_10.EPoint Center  
{ get; set; }
```

EPoint.CopyTo

Copies all the data of the current [EPoint](#) object into another [EPoint](#) object and returns it.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EPoint CopyTo(  
    Euresys.Open_eVision_2_10.EPoint other  
)
```

Parameters

other

Pointer to the [EPoint](#) object in which the current [EPoint](#) object data have to be copied.

Remarks

In case of a **NULL** pointer, a new [EPoint](#) object will be created and returned.

EPoint.Distance

Returns the distance between the addressed point and an [EPoint](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
float Distance(  
    Euresys.Open_eVision_2_10.EPoint point  
)  
  
float Distance(  
    Euresys.Open_eVision_2_10.ELine line,  
    bool segmentOnly  
)  
  
float Distance(  
    Euresys.Open_eVision_2_10.ECircle circle,  
    bool arcOnly  
)
```

Parameters

point

[EPoint](#) object with which to calculate the distance.

line

[ELine](#) object with which to calculate the distance.

segmentOnly

By default (**FALSE**), the line is not restricted to a segment.

circle

ECircle object with which to calculate the distance.

arcOnly

By default (**FALSE**), the circle is not restricted to an arc.

Remarks

Many EasyGauge members provide measurement result as a **EPoint** object (see **EPointGauge::Center**, **EPointGauge::GetMeasuredPoint**,...). The **EPoint** class has its own members to retrieve all the information pertaining to a point. Among them, the **Distance** method returns the distance between a point pair or between a point and a line segment, a circle arc or a rectangle.

EPoint.Dot

Compute the dot product of two **EPoint** object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float Dot(
    Euresys.Open_eVision_2_10.EPoint Point
)
```

Parameters

Point

Second factor of the dot product.

EPoint.EPoint

Constructs a **EPoint** object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EPoint(
)

void EPoint(
    float centerX,
    float centerY
)

void EPoint(
    Euresys.Open_eVision_2_10.EPoint other
)
```

Parameters

centerX

Center coordinates of the [EPoint](#) object.

centerY

Center coordinates of the [EPoint](#) object.

other

Another [EPoint](#) object to be copied in the new [EPoint](#) object.

EPoint.MidPoint

Returns the middle coordinate between this [EPoint](#) object and another [EPoint](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint MidPoint(
    Euresys.Open_eVision_2_10.EPoint Point
)
```

Parameters

Point

The other [EPoint](#) object.

EPoint.Modulus

Compute the euclidian modulus of a [EPoint](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Modulus (  
)
```

Remarks

Compute the squared root of the sum of the squared coordinates of an [EPoint](#). If the axis frame is orthogonal, this number is also the euclidian norm of the [EPoint](#).

EPoint.operator-

Subtracts from the current [EPoint](#) center coordinates the center coordinates of another [EPoint](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint operator- (  
    Euresys.Open_eVision_2_10.EPoint point  
)
```

Parameters

point

The other [EPoint](#) object.

EPoint.operator!=

Compares the current [EPoint](#) center coordinates with the center coordinates of another [EPoint](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool operator!=(
    Euresys.Open_eVision_2_10.EPoint point
)
```

Parameters

point

The other [EPoint](#) object.

Remarks

Returns **TRUE** if [EPoint::X](#) or [EPoint::Y](#) are respectively different.

EPoint.operator*

Multiplies the current [EPoint](#) center coordinates by a given multiplier.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint operator*(
    float scalar
)
```

Parameters

scalar

The multiplier.

EPoint.operator/

Divides the current [EPoint](#) center coordinates by a given divisor.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint operator/(  
    float scalar  
)
```

Parameters

scalar

The divisor.

EPoint.operator+

Adds to the current [EPoint](#) center coordinates the center coordinates of another [EPoint](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint operator+(  
    Euresys.Open_eVision_2_10.EPoint point  
)
```

Parameters

point

The other [EPoint](#) object.

EPoint.operator=

Copies all the data from another [EPoint](#) object into the current [EPoint](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint operator=(
    Euresys.Open_eVision_2_10.EPoint other
)
```

Parameters

other

[EPoint](#) object to be copied.

EPoint.operator==

Compares the current [EPoint](#) center coordinates with the center coordinates of another [EPoint](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool operator==(
    Euresys.Open_eVision_2_10.EPoint point
)
```

Parameters

point

The other [EPoint](#) object.

Remarks

Returns **TRUE** if both [EPoint::X](#) and [EPoint::Y](#) are respectively the same.

EPoint.Project

Compute the orthogonal projection of a [EPoint](#) on another shape.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint Project(
    Euresys.Open_eVision_2_10.ELine shape
)
Euresys.Open_eVision_2_10.EPoint Project(
    Euresys.Open_eVision_2_10.ECircle shape
)
```

Parameters

shape

Shape object to which point is projected

Remarks

Compute the orthogonal projection of a [EPoint](#) on another shape. This computation is only valid when the axis frame is orthogonal.

EPoint.Rotate

Returns another [EPoint](#) object containing the coordinated of the rotated point.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint Rotate(
    float angle
)
```

Parameters

angle

Rotation angle (in radians)

Remarks

Rotates a [EPoint](#) around the origin **(0, 0)** by an angle of **angle** radians. By definition, the smallest (in absolute value) rotation of the oriented X-Axis toward the oriented Y-Axis is chosen as the positive sense of rotation. In a direct frame, this is also the trigonometric sense (counter clockwise).

EPoint.SetCenterXY

Sets the center coordinates of a [EPoint](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetCenterXY(
    float centerX,
    float centerY
)
```

Parameters

centerX

Center coordinates of the [EPoint](#) object.

centerY

Center coordinates of the [EPoint](#) object.

EPoint.Square

Compute the sum of the squared coordinates of a [EPoint](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float Square(
)
```

Remarks

Compute the sum of the squared coordinates of a [EPoint](#). If the axis frame is orthogonal, this sum of squares is also the squared euclidian norm of the EPoint object.

EPoint.SquaredDistance

Compute the squared distance between two [EPoint](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float SquaredDistance(
    Euresys.Open_eVision_2_10.EPoint Point
)
```

Parameters

Point
Second [EPoint](#).

Remarks

Compute the sum of squared coordinates differences of two [EPoint](#). If the axis frame is orthogonal, this number is also the squared euclidian distance between two [EPoint](#).

EPoint.X

Abscissa (X coordinate) of the [EPoint](#) object

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float X  
{ get; }
```

EPoint.Y

Ordinate (Y coordinate) of the [EPoint](#) object

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Y  
{ get; }
```

4.143. EPointCloud Class

Represents a 3D point cloud.

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

[NumPoints](#)

Number of points in the [EPointCloud](#).

[PointsBuffer](#)

Retrieves a pointer to the internal points buffer.

Methods

AddPoint

Adds a point to the [EPointCloud](#).

AddPointCloud

Adds the points of another point cloud to [EPointCloud](#).

AddPoints

Adds a vector of points to the [EPointCloud](#).

Clear

Empties the [EPointCloud](#).

EPointCloud

Creates an [EPointCloud](#) object.

FillPointsBuffer

Copies an external points buffer into the internal points buffer.

GetPoint

Retrieves a point from the [EPointCloud](#).

Load

Loads the [EPointCloud](#). The given [ESerializer](#) must have been created for reading.

LoadPCD

Loads an [EPointCloud](#) stored in the PCD (Point Cloud Library) file format.
ASCII and binary formats are compatible.

operator=

Assignment operator.

Save

Saves the [EPointCloud](#). The given [ESerializer](#) must have been created for writing.

SavePCD

Saves the [EPointCloud](#) in the PCD (Point Cloud Library) file format. ASCII and binary formats are available.

Serialize

Serializes the [EPointCloud](#).

P

PointCloud.AddPoint

Adds a point to the [EPointCloud](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void AddPoint(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint point
)
```

Parameters

point

Point to add to the point cloud.

EPointCloud.AddPointCloud

Adds the points of another point cloud to [EPointCloud](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D


```
[C#]
void AddPointCloud(
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud cloud
)
```

Parameters

cloud

Point cloud whose points will be added to the point cloud.

EPointCloud.AddPoints

Adds a vector of points to the [EPointCloud](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void AddPoints(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint[] points
)
```

Parameters

points

Vector of points to add to the point cloud.

EPointCloud.Clear

Empties the [EPointCloud](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Clear(
)
```

EPointCloud.EPointCloud

Creates an [EPointCloud](#) object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void EPointCloud(
)
void EPointCloud(
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud other
)
```

Parameters

other

Reference to the [EPointCloud](#) used for the initialization.

EPointCloud.FillPointsBuffer

Copies an external points buffer into the internal points buffer.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void FillPointsBuffer(
    IntPtr pointsBuffer,
    int numPoints
)
```

Parameters

pointsBuffer

Address of the external points buffer.

numPoints

Number of points in the external points buffer.

Remarks

The buffer must contain points in the form of triplets of 32bits floats stored in the (X,Y,Z) order.

EPointCloud.GetPoint

Retrieves a point from the [EPointCloud](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DPoint GetPoint(
    uint index
)
```

Parameters

index

Index of the point to be retrieved.

EPointCloud.Load

Loads the [EPointCloud](#). The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Load(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is read from.

EPointCloud.LoadPCD

Loads an [EPointCloud](#) stored in the PCD (Point Cloud Library) file format. ASCII and binary formats are compatible.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void LoadPCD(
    string path
)
```

Parameters

path

The full path of the destination file.

Remarks

The PCD file format is documented here: http://pointclouds.org/documentation/tutorials/pcd_file_format.php.
Use `EPointCloud::SavePCD` to save to PCD file.

EPointCloud.NumPoints

Number of points in the [EPointCloud](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
int NumPoints
{ get; }
```

EPointCloud.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EPointCloud operator=(
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud other
)
```

Parameters

other

The [EPointCloud](#) object that should be copied.

EPointCloud.PointsBuffer

Retrieves a pointer to the internal points buffer.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
IntPtr PointsBuffer
{ get; }
```

EPointCloud.Save

Saves the [EPointCloud](#). The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is written to.

EPointCloud.SavePCD

Saves the [EPointCloud](#) in the PCD (Point Cloud Library) file format. ASCII and binary formats are available.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SavePCD(
    string path,
    bool binary
)
```

Parameters

path

The full path of the destination file.

binary

Optional parameter, activates the binary file format (default is false).

Remarks

The PCD file format is documented here: http://pointclouds.org/documentation/tutorials/pcd_file_format.php. Use [EPointCloud::LoadPCD](#) to load from PCD file.

EPointCloud.Serialize

Serializes the [EPointCloud](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
void Serialize(  
    Euresys.Open_eVision_2_10.ESerializer serializer  
)
```

Parameters

serializer

The [ESerializer](#) object that is read from or written to.

4.144. EPointCloudFactory Class

Manages a context for creating point clouds of specific shapes.

Namespace: Euresys.Open_eVision_2_10.Easy3D

Methods

[CreateCubicPointCloud](#)

Creates a point cloud in the shape of a cube.

[CreateRectangularPointCloud](#)

Creates a point cloud in the shape of a rectangular parallelepiped.

[CreateSphericPointCloud](#)

Creates a point cloud in the shape of a sphere.

P

PointCloudFactory.CreateCubicPointCloud

Creates a point cloud in the shape of a cube.

Namespace: Euresys.Open_eVision_2_10.Easy3D


```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EPointCloud CreateCubicPointCloud(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint center,
    float size,
    float roll,
    float pitch,
    float yaw,
    uint numSamples
)
```

Parameters

center

Center of the cube.

size

Edge size of the cube.

roll

Roll (rotation along the X axis) of the cube.

pitch

Pitch (rotation along the Y axis) of the cube.

yaw

Yaw (rotation along the Z axis) of the cube.

numSamples

Number of points along each edge of the cube.

EPointCloudFactory.CreateRectangularPointCloud

Creates a point cloud in the shape of a rectangular parallelepiped.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
Euresys.Open_eVision_2_10.Easy3D.EPointCloud CreateRectangularPointCloud(  
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint center,  
    float width,  
    float height,  
    float depth,  
    float roll,  
    float pitch,  
    float yaw,  
    uint numSamples  
)
```

Parameters

center

Center of the rectangular parallelepiped.

width

Width (size along the X axis before rotation) of the rectangular parallelepiped.

height

Height (size along the Y axis before rotation) of the rectangular parallelepiped.

depth

Depth (size along the Z axis before rotation) of the rectangular parallelepiped.

roll

Roll (rotation along the X axis) of the rectangular parallelepiped.

pitch

Pitch (rotation along the Y axis) of the rectangular parallelepiped.

yaw

Yaw (rotation along the Z axis) of the rectangular parallelepiped.

numSamples

Number of points along each edge of the rectangular parallelepiped.

EPointCloudFactory.CreateSphericPointCloud

Creates a point cloud in the shape of a sphere.

Namespace: Euresys.Open_eVision_2_10.Easy3D

[C#]

```
Euresys.Open_eVision_2_10.Easy3D.EPointCloud CreateSphericPointCloud(  
    ref Euresys.Open_eVision_2_10.Easy3D.E3DPoint center,  
    float radius,  
    int numCircles,  
    int numSamples  
)
```

Parameters

center

Center of the sphere.

radius

Radius of the sphere.

numCircles

Number of parallels and meridians to be rendered.

numSamples

Number of points along each meridian and parallel.

4.145. EPointCloudStatistics Class

Manages a context for retrieving statistics on an [EPointCloud](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

Methods

[GetPointCloudBounds](#)

Retrieves the bounds of an [EPointCloud](#).

GetPointCloudCentroid

PointCloudStatistics.GetPointCloudBounds

Retrieves the centroid (arithmetic mean position of all the points, also known as center of gravity or barycenter) of an [EPointCloud](#). It is possible to get the centroid of a sphere or a rectangle (rectangular parallelepiped) shape inside the point cloud. The sphere is defined by its center and radius, in the [EPointCloud](#) coordinate system. The 3D rectangle is defined by 3 ranges, in X, Y and Z axis, in the [EPointCloud](#) coordinate system. An exception will be thrown if no point is present in the shape or the point cloud.

Retrieves the bounds of an [EPointCloud](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void GetPointCloudBounds (
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud cloud,
    Euresys.Open_eVision_2_10.EFloatRange rangeX,
    Euresys.Open_eVision_2_10.EFloatRange rangeY,
    Euresys.Open_eVision_2_10.EFloatRange rangeZ
)
```

Parameters

cloud

Point cloud.

rangeX

Bounds of the point cloud along the X direction.

rangeY

Bounds of the point cloud along the Y direction.

rangeZ

Bounds of the point cloud along the Z direction.

EPointCloudStatistics.GetPointCloudCentroid

Retrieves the centroid (arithmetic mean position of all the points, also known as center of gravity or barycenter) of an [EPointCloud](#).

It is possible to get the centroid of a sphere or a rectangle (rectangular parallelepiped) shape inside the point cloud. The sphere is defined by its center and radius, in the [EPointCloud](#) coordinate system.

The 3D rectangle is defined by 3 ranges, in X, Y and Z axis, in the [EPointCloud](#) coordinate system.

An exception will be thrown if no point is present in the shape or the point cloud.

Namespace: Euresys.Open_eVision_2_10.Easy3D

[C#]

```
Euresys.Open_eVision_2_10.Easy3D.E3DPoint GetPointCloudCentroid(  
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud cloud  
)
```

```
Euresys.Open_eVision_2_10.Easy3D.E3DPoint GetPointCloudCentroid(  
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud cloud,  
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint sphereCenter,  
    float sphereRadius  
)
```

```
Euresys.Open_eVision_2_10.Easy3D.E3DPoint GetPointCloudCentroid(  
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud cloud,  
    Euresys.Open_eVision_2_10.EFloatRange rangeX,  
    Euresys.Open_eVision_2_10.EFloatRange rangeY,  
    Euresys.Open_eVision_2_10.EFloatRange rangeZ  
)
```

Parameters

cloud

Point cloud.

sphereCenter

The position of the center of the sphere.

sphereRadius

The radius of the sphere.

rangeX

The bounds along the X direction.

rangeY

The bounds along the Y direction.

rangeZ

The bounds along the Z direction.

4.146. EPointCloudToZMapConverter Class

Computes a [EZMap](#) from a [EPointCloud](#). The value of the pixels of the ZMap are the distance between the 3D points and the reference plane.

All 3D points under the reference plane are discarded.

Various options can be set with methods [EPointCloudToZMapConverter](#), [EPointCloudToZMapConverter::SetFillMode](#), [EPointCloudToZMapConverter](#), [EPointCloudToZMapConverter](#), [EPointCloudToZMapConverter...](#)

When the conversion is called without defining specific parameters, the algorithm uses the following options:

- The reference plane is the horizontal plane.
- The orientation vector is selected automatically.
- The origin is set as the lowest left position of the projected point cloud on the reference plane.
- The resolution (the dimensions of the Z map) is estimated to have approximately one Point Cloud point per ZMap pixels.
- The scale is calculated from the point cloud ranges and the estimated resolution.
- The fill mode is enabled and the method is set to 'EFillUndefinedPixelsDirection_Local' (see method [EDepthMap8::FillUndefinedPixels](#)).

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

Extension

Sets a metric value used to enlarge the point cloud 3D domain. That value affects X,Y and Z directions and can be used to generate an [EZMap](#) with borders of undefined pixels. Default value is **0**, which means a ZMap without border.

FillUndefinedPixelsDirection

Gets the undefined pixel fill direction (see [EFillUndefinedPixelsDirection](#)).

[FillUndefinedPixelsMethod](#)

Gets the undefined pixel fill method (see [EFillUndefinedPixelsMethod](#)).

[MapHeight](#)

Gets the required height (number of pixels) of the generated [EZMap](#).
By default, the required size is not set.

[MapWidth](#)

Gets the required width (number of pixels) of the generated [EZMap](#).
By default, the required size is not set.

[MapXResolution](#)

Gets the resolution of the [EZMap](#) pixels along the X axis.

[MapYResolution](#)

Gets the resolution of the [EZMap](#) pixels along the Y axis.

[MapZResolution](#)

Gets/sets the [EZMap](#) Z resolution, in world space units per gray value.
The resolution is used to compute the transformation of the world Z position to an integer 8, 16 or 32 bits pixel value.

[OrientationVector](#)

Sets an explicit orientation for the [EZMap](#).
Overrides the orientation mode given by the method [EPointCloudToZMapConverter](#).

[OrientationVectorMode](#)

Chooses the [EZMap](#) orientation from a list of pre-defined axis, automatic mode or user defined vector.
Use [EPointCloudToZMapConverter](#) to set an explicit orientation vector for the ZMap.

Origin

Chooses the [EZMap](#) origin. It is the 3D world position used as origin of the ZMap upper left pixel (0,0).

ReferencePlane

Sets the [E3DPlane](#) reference plane.
The resulting [EZMap](#) is the distance of the 3D points above that plane.
3D points below the reference plane are discarded.

ReferencePlaneMode

Sets an axis aligned reference plane.
Overrides the explicit reference plane given by the method [EPointCloudToZMapConverter](#).

WorldToZMapTransform

Explicitly sets the world to ZMap transformation.
"SetWorldToZMapTransform" overrides the settings done by [EPointCloudToZMapConverter](#), [EPointCloudToZMapConverter](#) and [EPointCloudToZMapConverter](#).
That [E3DTransformMatrix](#) transform expresses how the world positions are transformed to the [EZMap](#) space.
The matrix must be a rigid transformation (translation and rotation only).
The resolution and scales of the ZMap are defined by [EPointCloudToZMapConverter](#), [EPointCloudToZMapConverter](#) and [EPointCloudToZMapConverter](#) methods.

ZMapToWorldTransform

Methods

Explicitly sets the ZMap to World transformation. "SetZMapToWorldTransform" overrides the settings done by [EPointCloudToZMapConverter](#), [EPointCloudToZMapConverter](#) and [EPointCloudToZMapConverter](#). That [E3DTransformMatrix](#) transform expresses how the [EZMap](#) positions are transformed to the World space. The matrix must be a rigid transformation (translation and rotation only). The resolution and scales of the World are defined by [EPointCloudToZMapConverter](#), [EPointCloudToZMapConverter](#) and [EPointCloudToZMapConverter](#) methods.

Convert

Computes a EZMap from a world space EPointCloud. The value of the pixels of the ZMap are the distance between the 3D points and the reference plane. Various options can be set with methods [EPointCloudToZMapConverter](#), [EPointCloudToZMapConverter](#), [EPointCloudToZMapConverter::SetMapSize](#), ...

EnableFillMode

Enables or disables fill mode. Fill mode parameters are defined by method [EPointCloudToZMapConverter::SetFillMode](#). Fill mode is enabled by default. If fill mode is disabled, undefined pixels may remain in the [EZMap](#).

EPointCloudToZMapConverter

Creates a [EPointCloudToZMapConverter](#) object.

IsFillModeEnabled

Tells if the fill mode is enabled or not. Use [EPointCloudToZMapConverter::EnableFillMode](#) to toggle the fill mode and [EPointCloudToZMapConverter::SetFillMode](#) to set the filling parameters.

Load	Loads the converter configuration. The given ESerializer must have been created for reading.
operator=	Assignment operator
operator==	Comparison operator
Save	Saves the converter configuration. The given ESerializer must have been created for writing.
SetFillMode	Inpainting options used to fill the "holes" in the EZMap . A hole exists when no 3D point is projected at that pixel position in the ZMap.
SetMapSize	Sets the required size of the generated EZMap ; expressed in number of pixels for width and height dimensions. By default, the required size is not set.
SetMapXYResolution	Sets the resolution (possibly anisotropic) of the EZMap pixels along the X and Y axes, in world space units per pixel (e.g mm/pixel). The resolution is used to compute the ZMap size (width and height), depending on the projected point cloud on the reference plane.
UnsetMapSize	Unsets the resolution. Lets the conversion decides for the optimum resolution, depending on the projected point cloud and pixel scale.

[UnsetMapXYResolution](#)

Unsets the X and Y resolutions. Lets the conversion decide the optimal resolution, depending on the projected point cloud and [EZMap](#) size.

[UnsetMapZResolution](#)

Unsets the ZMap Z resolution. Lets the conversion decide the optimal Z resolution.

[UnsetOrigin](#)

Lets the conversion process decides for the [EZMap](#) origin position (based on projected point cloud on the reference plane). Use [EPointCloudToZMapConverter](#) to enable and choose the ZMap origin.

[UnsetWorldToZMapTransform](#)

┌ Disables the explicit world to ZMap transformation, set with
└ [EPointCloudToZMapConverter](#).

P

PointCloudToZMapConverter.Convert

Computes a EZMap from a world space EPointCloud. The value of the pixels of the ZMap are the distance between the 3D points and the reference plane. Various options can be set with methods [EPointCloudToZMapConverter](#), [EPointCloudToZMapConverter](#), [EPointCloudToZMapConverter::SetMapSize](#), ...

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Convert(
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud cloud,
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 zmap
)
```

```
void Convert(  
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud cloud,  
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 zmap  
)  
  
void Convert(  
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud cloud,  
    Euresys.Open_eVision_2_10.Easy3D.EZMap32f zmap  
)
```

Parameters

cloud

The input 3D point cloud.

zmap

The generated ZMap in 8, 16 or 32 bits format.

EPointCloudToZMapConverter.EnableFillMode

Enables or disables fill mode. Fill mode parameters are defined by method

[EPointCloudToZMapConverter::SetFillMode](#).

Fill mode is enabled by default. If fill mode is disable, undefined pixels may remain in the [EZMap](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
void EnableFillMode(  
    bool state  
)
```

Parameters

state

Set to true to enable fill mode.

EPointCloudToZMapConverter.EPointCloudToZMapConverter

Creates a [EPointCloudToZMapConverter](#) object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void EPointCloudToZMapConverter (
)
void EPointCloudToZMapConverter (
    Euresys.Open_eVision_2_10.Easy3D.EPointCloudToZMapConverter other
)
```

Parameters

other

Reference to the [EPointCloudToZMapConverter](#) object used for the initialization.

EPointCloudToZMapConverter.Extension

Sets a metric value used to enlarge the point cloud 3D domain.

That value affects X,Y and Z directions and can be used to generate an [EZMap](#) with borders of undefined pixels. Default value is **0**, which means a ZMap without border.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
float Extension
{ get; set; }
```

EPointCloudToZMapConverter.FillUndefinedPixelsDirection

Gets the undefined pixel fill direction (see [EFillUndefinedPixelsDirection](#)).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EFillUndefinedPixelsDirection
FillUndefinedPixelsDirection
{ get; }
```

EPointCloudToZMapConverter.FillUndefinedPixelsMethod

Gets the undefined pixel fill method (see [EFillUndefinedPixelsMethod](#)).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EFillUndefinedPixelsMethod FillUndefinedPixelsMethod
{ get; }
```

EPointCloudToZMapConverter.IsFillModeEnabled

Tells if the fill mode is enabled or not.

Use [EPointCloudToZMapConverter::EnableFillMode](#) to toggle the fill mode and [EPointCloudToZMapConverter::SetFillMode](#) to set the filling parameters.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool IsFillModeEnabled(
)
```

EPointCloudToZMapConverter.Load

Loads the converter configuration. The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Load(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

Pointer to the [ESerializer](#) created for reading.

EPointCloudToZMapConverter.MapHeight

Gets the required height (number of pixels) of the generated [EZMap](#).
By default, the required size is not set.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
int MapHeight
```

```
{ get; }
```

EPointCloudToZMapConverter.MapWidth

Gets the required width (number of pixels) of the generated [EZMap](#).
By default, the required size is not set.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
int MapWidth
```

```
{ get; }
```

EPointCloudToZMapConverter.MapXResolution

Gets the resolution of the [EZMap](#) pixels along the X axis.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
float MapXResolution
```

```
{ get; }
```


EPointCloudToZMapConverter.MapYResolution

Gets the resolution of the [EZMap](#) pixels along the Y axis.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float MapYResolution  
{ get; }
```

EPointCloudToZMapConverter.MapZResolution

Gets/sets the [EZMap](#) Z resolution, in world space units per gray value.
The resolution is used to compute the transformation of the world Z position to an integer 8, 16 or 32 bits pixel value.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float MapZResolution  
{ get; set; }
```

EPointCloudToZMapConverter.operator=

Assignment operator

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EPointCloudToZMapConverter operator=(
    Euresys.Open_eVision_2_10.Easy3D.EPointCloudToZMapConverter other
)
```

Parameters

other

Reference to the [EPointCloudToZMapConverter](#) object used for the assignment.

EPointCloudToZMapConverter.operator==

Comparison operator

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool operator==(
    Euresys.Open_eVision_2_10.Easy3D.EPointCloudToZMapConverter other
)
```

Parameters

other

Reference to the [EPointCloudToZMapConverter](#) object used for the comparison.

EPointCloudToZMapConverter.OrientationVector

Sets an explicit orientation for the [EZMap](#).
Overrides the orientation mode given by the method [EPointCloudToZMapConverter](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
Euresys.Open_eVision_2_10.Easy3D.E3DPoint OrientationVector  
  
{ get; set; }
```

Remarks

The direction should be an [E3DPoint](#) representing the expected direction of the X (width) axis of the ZMap. That direction will be used after projection on the reference plane normal. That direction must NOT be aligned with the reference plane normal.

EPointCloudToZMapConverter.OrientationVectorMode

Chooses the [EZMap](#) orientation from a list of pre-defined axis, automatic mode or user defined vector. Use [EPointCloudToZMapConverter](#) to set an explicit orientation vector for the ZMap.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
Euresys.Open_eVision_2_10.Easy3D.EZMapOrientationVectorMode OrientationVectorMode  
  
{ get; set; }
```

Remarks

Choose between Automatic mode (default), world space axis or explicit user defined vector (see [EZMapOrientationVectorMode](#)).

EPointCloudToZMapConverter.Origin

Chooses the [EZMap](#) origin. It is the 3D world position used as origin of the ZMap upper left pixel (0,0).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DPoint Origin
{ get; set; }
```

Remarks

That position will be projected on the reference plane.

To let the conversion chooses for the origin, call [EPointCloudToZMapConverter::UnsetOrigin](#).

EPointCloudToZMapConverter.ReferencePlane

Sets the [E3DPlane](#) reference plane.
The resulting [EZMap](#) is the distance of the 3D points above that plane.
3D points below the reference plane are discarded.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DPlane ReferencePlane
{ get; set; }
```

EPointCloudToZMapConverter.ReferencePlaneMode

Sets an axis aligned reference plane.
Overrides the explicit reference plane given by the method [EPointCloudToZMapConverter](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
Euresys.Open_eVision_2_10.Easy3D.EZMapReferencePlaneMode ReferencePlaneMode  
  
{ get; set; }
```

Remarks

Choose between X, Y or Z reference plane (see [EZMapReferencePlaneMode](#)).
The plane offset is set automatically on the point cloud lowest 3D point.

EPointCloudToZMapConverter.Save

Saves the converter configuration. The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
void Save(  
    Euresys.Open_eVision_2_10.ESerializer serializer  
)
```

Parameters

serializer

Pointer to the [ESerializer](#) created for writing.

EPointCloudToZMapConverter.SetFillMode

Inpainting options used to fill the "holes" in the [EZMap](#). A hole exists when no 3D point is projected at that pixel position in the ZMap.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetFillMode(
    Euresys.Open_eVision_2_10.Easy3D.EFillUndefinedPixelsDirection direction,
    Euresys.Open_eVision_2_10.Easy3D.EFillUndefinedPixelsMethod method
)
```

Parameters

direction

Direction in which the undefined pixels are filled in a depthmap from [EFillUndefinedPixelsDirection](#)

method

Which values used to fill the undefined pixels in a depthmap from [EFillUndefinedPixelsMethod](#)

EPointCloudToZMapConverter.SetMapSize

Sets the required size of the generated [EZMap](#); expressed in number of pixels for width and height dimensions. By default, the required size is not set.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetMapSize(
    int width,
    int height
)
```

Parameters

width

The required width for the Generated ZMap.

height

The required height for the Generated ZMap.

EPointCloudToZMapConverter.SetMapXYResolution

Sets the resolution (possibly anisotropic) of the [EZMap](#) pixels along the X and Y axes, in world space units per pixel (e.g mm/pixel).
The resolution is used to compute the ZMap size (width and height), depending on the projected point cloud on the reference plane.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetMapXYResolution(
    float resolution
)
void SetMapXYResolution(
    float resolutionX,
    float resolutionY
)
```

Parameters

resolution

The resolution for the isotropic case.

resolutionX

The resolution for the X axis.

resolutionY

The resolution for the Y axis.

Remarks

The isotropic scale, for X and Y axis is in metric world units.

EPointCloudToZMapConverter.UnsetMapSize

Unsets the resolution. Lets the conversion decides for the optimum resolution, depending on the projected point cloud and pixel scale.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void UnsetMapSize (
)
```

EPointCloudToZMapConverter.UnsetMapXYResolution

Unsets the X and Y resolutions. Lets the conversion decide the optimal resolution, depending on the projected point cloud and [EZMap](#) size.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void UnsetMapXYResolution (
)
```

EPointCloudToZMapConverter.UnsetMapZResolution

Unsets the ZMap Z resolution. Lets the conversion decide the optimal Z resolution.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void UnsetMapZResolution (
)
```


EPointCloudToZMapConverter.UnsetOrigin

Lets the conversion process decides for the [EZMap](#) origin position (based on projected point cloud on the reference plane).

Use [EPointCloudToZMapConverter](#) to enable and choose the ZMap origin.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void UnsetOrigin(  
)
```

EPointCloudToZMapConverter.UnsetWorldToZMapTransform

Disables the explicit world to ZMap transformation, set with [EPointCloudToZMapConverter](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void UnsetWorldToZMapTransform(  
)
```

EPointCloudToZMapConverter.WorldToZMapTransform

Explicitly sets the world to ZMap transformation.

"SetWorldToZMapTransform" overrides the settings done by [EPointCloudToZMapConverter](#), [EPointCloudToZMapConverter](#) and [EPointCloudToZMapConverter](#).

That [E3DTransformMatrix](#) transform expresses how the world positions are transformed to the [EZMap](#) space. The matrix must be a rigid transformation (translation and rotation only).

The resolution and scales of the ZMap are defined by [EPointCloudToZMapConverter](#), [EPointCloudToZMapConverter](#) and [EPointCloudToZMapConverter](#) methods.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix WorldToZMapTransform
    { get; set; }
```

EPointCloudToZMapConverter.ZMaptoWorldTransform

Explicitly sets the ZMap to World transformation.

"SetZMaptoWorldTransform" overrides the settings done by [EPointCloudToZMapConverter](#), [EPointCloudToZMapConverter](#) and [EPointCloudToZMapConverter](#).

That [E3DTransformMatrix](#) transform expresses how the [EZMap](#) positions are transformed to the World space. The matrix must be a rigid transformation (translation and rotation only).

The resolution and scales of the World are defined by [EPointCloudToZMapConverter](#), [EPointCloudToZMapConverter](#) and [EPointCloudToZMapConverter](#) methods.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix ZMaptoWorldTransform
    { get; set; }
```

4.147. EPointGauge Class

Manages a point location gauge.

Base Class: [EPointShape](#)

Namespace: Euresys.Open_eVision_2_10

Properties

Active	Sets the flag indicating whether the gauge is active or not.
Center	Center coordinates of a EPointGauge object.
HVConstraint	Status of the restriction on the orientation of the point location gauge or model fitting sample paths.
MinAmplitude	Offset added to the Threshold when a peak is to be detected.
MinArea	Minimum area value.
NumMeasuredPoints	Number of edge-crossing points along the point location gauge.
RectangularSamplingArea	Flag indicating whether the sampling area remains a rectangle when rotated, instead of becoming a parallelogram.

Smoothing

Number of pixels used for the low-pass filtering operation.

Thickness

Number of parallel segments used to extract the data profile.

Threshold

Threshold level used to delimit significant peaks in the data profile.

Tolerance

Half length of the point location gauge.

ToleranceAngle

Rotation angle of the point location gauge.

TransitionChoice

Transition choice.

TransitionIndex

Index (from **0** on) of the transition to be retained when the transition choice parameter is set to [NthFromBegin](#) or [NthFromEnd](#).

TransitionType

Transition type.

Type

Shape type.

Valid

M Flag indicating if at least one valid transition has been found.

e

thods

CopyTo

Copies all the data of the current EPointGauge object into another EPointGauge object, and returns it.

Drag

Moves a handle to a new position and updates the position parameters of the gauge.

Draw

Draws a graphical representation of a point location or model fitting gauge, as defined by [EDrawingMode](#).

DrawWithCurrentPen

Draws a graphical representation of a point location or model fitting gauge, as defined by [EDrawingMode](#).

EPointGauge

Constructs a point measurement context.

GetMeasuredPeak

Returns information pertaining to the derivative peak associated with the specified edge-crossing point, such as its area, amplitude, start, length and center.

GetMeasuredPoint

Returns the coordinates of an edge-crossing point, measured along the unique sample path of the point location gauge.

HitTest

Checks whether the cursor is positioned over a handle (**TRUE**) or not (**FALSE**).

Measure

Triggers the point location or the model fitting operation.

operator=

Copies all the data from another EPointGauge object into the current EPointGauge object

Plot	Draws the profile that is crossed by a point location gauge, as defined by EPlotItem .
PlotWithCurrentPen	Draws the profile that is crossed by a point location gauge, as defined by EPlotItem .
Process	Triggers the process pertaining to a shape or gauge and all the daughter gauges attach to it.
SetCenterXY	Sets the center coordinates of a EPointGauge object.
SetTolerances	<div style="border-left: 1px solid black; padding-left: 5px;"> Sets the half length and the rotation angle of the point location gauge. </div>

PointGauge.Active

Sets the flag indicating whether the gauge is active or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
override bool Active
{ get; set; }
```

Remarks

When complex gauging is required, several gauges can be grouped together. Applying [EPointGauge::Process](#) to the mother gauge or shape triggers the measurement of the whole. Only the active gauges will participate in the process. By default, the gauge is active (**TRUE**).

EPointGauge.Center

Center coordinates of a [EPointGauge](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
override Euresys.Open_eVision_2_10.EPoint Center
    { get; set; }
```

EPointGauge.CopyTo

Copies all the data of the current EPointGauge object into another EPointGauge object, and returns it.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPointGauge CopyTo(
    Euresys.Open_eVision_2_10.EPointGauge other,
    bool recursive
)
```

Parameters

other

Pointer to the EPointGauge object in which the current EPointGauge object data have to be copied.

recursive

TRUE if the children gauges have to be copied as well, **FALSE** otherwise.

Remarks

In case of a **NULL** pointer, a new EPointGauge object will be created and returned.

EPointGauge.Drag

Moves a handle to a new position and updates the position parameters of the gauge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Drag(
    int x,
    int y
)
```

Parameters

x
Cursor current X coordinate.

y
Cursor current Y coordinate.

EPointGauge.Draw

Draws a graphical representation of a point location or model fitting gauge, as defined by [EDrawingMode](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)
```



```

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)

void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)

```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingMode

Indicates how the point location or model fitting gauge must be displayed, as defined by [EDrawingMode](#).

daughters

TRUE if the daughters gauges are to be displayed also.

color

The color in which to draw the overlay.

EPointGauge.DrawWithCurrentPen

Draws a graphical representation of a point location or model fitting gauge, as defined by [EDrawingMode](#).

Namespace: Euresys.Open_eVision_2_10

```

[C#]

void DrawWithCurrentPen(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)

```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingMode

Indicates how the point location or model fitting gauge must be displayed, as defined by [EDrawingMode](#).

daughters

TRUE if the daughters gauges are to be displayed also.

EPointGauge.EPointGauge

Constructs a point measurement context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EPointGauge (
)
void EPointGauge (
    float centerX,
    float centerY
)
void EPointGauge (
    Euresys.Open_eVision_2_10.EPointGauge other
)
```

Parameters

centerX

Point X coordinate.

centerY

Point Y coordinate.

other

Another EPointGauge object to be copied in the new EPointGauge object.

Remarks

With the default constructor, all the parameters are initialized to their respective default values. With the copy constructor, the constructed point measurement context is based on a pre-existing [EPointGauge](#) object. The gauge children are also copied. Hierarchy copying through a copy constructor is always recursive. To disable this recursion, use instead the [EPointGauge::CopyTo](#) method.

EPointGauge.GetMeasuredPeak

Returns information pertaining to the derivative peak associated with the specified edge-crossing point, such as its area, amplitude, start, length and center.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPeak GetMeasuredPeak (
    uint index
)
```

Parameters

index

Index of the edge-crossing point along the probed line segment, between **0** and [EPointGauge::NumMeasuredPoints](#) (excluded).

Remarks

If **index** is left unchanged from its default value (i.e. **~0 = 0xFFFFFFFF**), the peak associated to the default edge-crossing point is inspected. This default point is chosen according to the transition choice parameter that is managed by the [EPointGauge::TransitionChoice](#) property.

Note. For this method to succeed, it is necessary to previously call [EPointGauge::Measure](#).

EPointGauge.GetMeasuredPoint

Returns the coordinates of an edge-crossing point, measured along the unique sample path of the point location gauge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint GetMeasuredPoint(
    uint index
)
```

Parameters

index

Index of the edge-crossing point along the probed line segment, between **0** and [EPointGauge::NumMeasuredPoints](#) (excluded).

Remarks

These coordinates pertain to the World space; they are expressed in the reference frame to which the current [EPointGauge](#) object belongs. An [EPointGauge](#) object features only one sample path, which contrasts with the other kinds of gauges. The argument **index** specifies the index of the edge-crossing point that is considered along this unique sample path. If **index** is left unchanged from its default value (i.e. **~0= 0xFFFFFFFF**), the default edge-crossing point is inspected. This default point is chosen according to the transition choice parameter that is managed by the [EPointGauge::TransitionChoice](#) property.

Note. For this method to succeed, it is necessary to previously call [EPointGauge::Measure](#).

EPointGauge.HitTest

Checks whether the cursor is positioned over a handle (**TRUE**) or not (**FALSE**).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool HitTest(
    bool daughters
)
```

Parameters

daughters

TRUE if the daughters gauges handles have to be considered as well.

EPointGauge.HVConstraint

Status of the restriction on the orientation of the point location gauge or model fitting sample paths.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool HVConstraint  
    { get; set; }
```

Remarks

Sample paths are the point location gauges placed along the model to be fitted.

EPointGauge.Measure

Triggers the point location or the model fitting operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Measure(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage  
)
```

Parameters

sourceImage

Pointer to the source image.

Remarks

When this method is called, and if not enough valid points are detected, then the method returns directly, and the measured gauge is set to the nominal parameters.

EPointGauge.MinAmplitude

Offset added to the **Threshold** when a peak is to be detected.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint MinAmplitude
{ get; set; }
```

Remarks

When analyzing a derivative profile, a peak is made up of consecutive pixel values above **Threshold**. To detect weak [strong] transitions, lower [raise] the **Threshold** value. To avoid interference of noise, an additional parameter is provided. The **MinAmplitude** parameter is an offset added to **Threshold** when a peak is to be detected. When the pixel values of the derivative profile do not reach **Threshold + MinAmplitude**, the peak is not taken into account. Anyway, when a peak is taken into account, all the pixels with values above **Threshold** are considered (for more accuracy). Setting the **MinAmplitude** value to **0** merely cancels its effect.

EPointGauge.MinArea

Minimum area value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint MinArea
```

```
{ get; set; }
```

Remarks

A transition is detected if its derivative peak reaches **Threshold + MinAmplitude** value, and then declared valid if the area between the peak curve and the horizontal at level **Threshold** reaches the **MinArea** value.

EPointGauge.NumMeasuredPoints

Number of edge-crossing points along the point location gauge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
uint NumMeasuredPoints  
  
{ get; }
```

EPointGauge.operator=

Copies all the data from another EPointGauge object into the current EPointGauge object

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
Euresys.Open_eVision_2_10.EPointGauge operator=(  
    Euresys.Open_eVision_2_10.EPointGauge other  
)
```

Parameters

other

EPointGauge object to be copied

EPointGauge.Plot

Draws the profile that is crossed by a point location gauge, as defined by [EPlotItem](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Plot(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EPlotItem drawItems,
    float width,
    float height,
    float originX,
    float originY
)

void Plot(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.EPlotItem drawItems,
    float width,
    float height,
    float originX,
    float originY
)

void Plot(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.EPlotItem drawItems,
    float width,
    float height,
    float originX,
    float originY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawItems

Boolean combination of [EPlotItem](#) members, that indicates which items must be displayed.

width

Width of the plot.

height

Height of the plot.

originX

Origin point coordinates of the plot along the X axis.

originY

Origin point coordinates of the plot along the Y axis.

color

The color in which to draw the overlay.

EPointGauge.PlotWithCurrentPen

Draws the profile that is crossed by a point location gauge, as defined by [EPlotItem](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void PlotWithCurrentPen(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EPlotItem drawItems,
    float width,
    float height,
    float originX,
    float originY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawItems

Boolean combination of [EPlotItem](#) members, that indicates which items must be displayed.

width

Width of the plot.

height

Height of the plot.

originX

Origin point coordinates of the plot along the X axis.

originY

Origin point coordinates of the plot along the Y axis.

EPointGauge.Process

Triggers the process pertaining to a shape or gauge and all the daughter gauges attach to it.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Process (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    bool daughters
)
```

Parameters

sourceImage

Pointer to the source image.

daughters

Flag indicating whether the daughters shapes inherit of the same behavior.

Remarks

When complex gauging is required, several gauges can be grouped together. Applying **Process** to the mother gauge or shape triggers the measurement of the whole. Only the active gauges will participate in the process.

EPointGauge.RectangularSamplingArea

Flag indicating whether the sampling area remains a rectangle when rotated, instead of becoming a parallelogram.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool RectangularSamplingArea
    { get; set; }
```

Remarks

By default, this flag is set to **TRUE**: the sampling area always remains a rectangle. Setting this property is only useful when the thickness transition parameter is greater than 1. In fact, when thickness transition parameter is equal to 1, rectangle and parallelogram reduce to the same segment.

EPointGauge.SetCenterXY

Sets the center coordinates of a [EPointGauge](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetCenterXY(
    float centerX,
    float centerY
)
```

Parameters

centerX

Center coordinates of the [EPointGauge](#) object.

centerY

Center coordinates of the [EPointGauge](#) object.

EPointGauge.SetTolerances

Sets the half length and the rotation angle of the point location gauge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetTolerances (
    float tolerance,
    float angle
)
```

Parameters

tolerance

Half length of the point location gauge. The default value is **10**.

angle

Rotation angle of the point location gauge. The default value is **0**.

Remarks

By default, the point location gauge length value is 20 (2x10), which means 20 pixels when the field of view is not calibrated and 20 "units" in case of a calibrated field of view. The sign of the rotation angle depends whether the field of view is calibrated or not. * When the field of view is calibrated, the coordinate system is said to be direct, the abscissa extends rightwards and the ordinate extends upwards. In this case, an anticlockwise rotation leads to a positive angle value. * When the field of view is not calibrated, the coordinate system is said to be inverse, the abscissa extends rightwards and the ordinate extends downwards. In this case, a clockwise rotation leads to a positive angle value.

EPointGauge.Smoothing

Number of pixels used for the low-pass filtering operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint Smoothing
```

```
{ get; set; }
```

Remarks

To reduce the effect of noise, the profile data can be low-pass filtered along the point location gauge direction.

EPointGauge.Thickness

Number of parallel segments used to extract the data profile.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint Thickness
```

```
{ get; set; }
```

Remarks

To reduce the effect of noise and/or strengthen a transition, several parallel profiles can be accumulated.

EPointGauge.Threshold

Threshold level used to delimit significant peaks in the data profile.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint Threshold
```

```
{ get; set; }
```

Remarks

When analyzing a derivative profile, a peak is made up of consecutive pixel values above **Threshold**. To detect weak [strong] transitions, lower [raise] the **Threshold** value. To avoid interference of noise, an additional parameter is provided. The **MinAmplitude** parameter is an offset added to **Threshold** when a peak is to be detected. When the pixel values of the derivative profile do not reach **Threshold + MinAmplitude**, the peak is not taken into account. Anyway, when a peak is taken into account, all the pixels with values above **Threshold** are considered (for more accuracy). Setting the **MinAmplitude** value to **0** merely cancels its effect.

EPointGauge.Tolerance

Half length of the point location gauge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float Tolerance
```

```
{ get; set; }
```

Remarks

By default, the length of the point location gauge is 20 (2x10), which means 20 pixels when the field of view is not calibrated, and 20 physical units in case of a calibrated field of view.

EPointGauge.ToleranceAngle

Rotation angle of the point location gauge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ToleranceAngle
```

```
{ get; set; }
```

Remarks

By default, the rotation angle of the point location gauge is **0**. The sign of the rotation angle depends whether the field of view is calibrated or not. * When the field of view is calibrated, the coordinate system is said to be direct, the abscissa extends rightwards and the ordinate extends upwards. In this case, an anticlockwise rotation leads to a positive angle value. * When the field of view is not calibrated, the coordinate system is said to be inverse, the abscissa extends rightwards and the ordinate extends downwards. In this case, a clockwise rotation leads to a positive angle value.

EPointGauge.TransitionChoice

Transition choice.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.ETransitionChoice TransitionChoice
```

```
{ get; set; }
```

Remarks

Several peaks may be detected along a point location gauge. This parameter helps to select the desired transition. In case of [NthFromBegin](#) or [NthFromEnd](#) transition choice, set [EPointGauge::TransitionIndex](#) to specify the desired transition. By default, the selected transition corresponds to the one with the largest amplitude ([LargestAmplitude](#)).

EPointGauge.TransitionIndex

Index (from **0** on) of the transition to be retained when the transition choice parameter is set to [NthFromBegin](#) or [NthFromEnd](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint TransitionIndex  
  
{ get; set; }
```

Remarks

Several peaks may be detected along a point location gauge. This parameter helps to select the desired transition. By default, the first transition is retained (the index value is **0**).

EPointGauge.TransitionType

Transition type.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.ETransitionType TransitionType  
  
{ get; set; }
```

Remarks

The type of a transition tells whether it crosses increasing or decreasing gray-level values. This helps discriminate between nearby edges of an object. By default, the searched transition type is indifferently a black to white or a white to black transition ([BwOrWb](#)).

EPointGauge.Type

Shape type.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
override Euresys.Open_eVision_2_10.EShapeType Type
{ get; }
```

EPointGauge.Valid

Flag indicating if at least one valid transition has been found.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool Valid
{ get; }
```

Remarks

A **FALSE** value means that no measurement has been performed. A **TRUE** value means that a transition was found along the sample path defined by the [EPointGauge](#), and thus a point was measured.

4.148. EPointShape Class

Manages a point shape context.

Base Class: [EShape](#)

Derived Class(es): [EPatternFinder](#) [EPointGauge](#)

Namespace: Euresys.Open_eVision_2_10

Properties

Center | Center coordinates of a [EPointShape](#) object.

CenterX | Abscissa of the origin point of the frame.

CenterY | Ordinate of the origin point of the frame.

Type | **M** Shape type.

e

Methods

Closest | Find the daughter shape that is the closest to this shape. To retrieve the closest shape, use [EShape::ClosestShape](#).

CopyTo | Copies all the data of the current [EPointShape](#) object into another [EPointShape](#) object, and returns it.

Drag | Moves a handle to a new position and updates the position parameters of the gauge.

Draw	Draws a graphical representation of a shape, as defined by EDrawingMode .
DrawWithCurrentPen	Draws a graphical representation of a shape, as defined by EDrawingMode .
HitTest	Checks if there is a handle under the cursor.
operator!=	Compares the instance another EPointShape object and returns TRUE if they are not identical.
operator=	Copies all the data from another EPointShape object into the current EPointShape object
operator==	Compares the instance another EPointShape object and returns TRUE if they are identical.
SetCenterXY	<div style="border-left: 1px solid black; padding-left: 5px;"> Sets the center coordinates of a EPointShape object. </div>

P

ointShape.Center

Center coordinates of a [EPointShape](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
virtual Euresys.Open_eVision_2_10.EPoint Center
    { get; set; }
```

EPointShape.CenterX

Abscissa of the origin point of the frame.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float CenterX
    { get; }
```

EPointShape.CenterY

Ordinate of the origin point of the frame.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float CenterY
    { get; }
```

EPointShape.Closest

Find the daughter shape that is the closest to this shape. To retrieve the closest shape, use [EShape::ClosestShape](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Closest(
)
```

EPointShape.CopyTo

Copies all the data of the current EPointShape object into another EPointShape object, and returns it.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPointShape CopyTo(
    Euresys.Open_eVision_2_10.EPointShape other,
    bool recursive
)
```

Parameters

other

Pointer to the EPointShape object in which the current EPointShape object data have to be copied.

recursive

TRUE if the children gauges have to be copied as well, **FALSE** otherwise.

Remarks

In case of a **NULL** pointer, a new EPointShape object will be created and returned.

EPointShape.Drag

Moves a handle to a new position and updates the position parameters of the gauge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Drag(
    int n32CursorX,
    int n32CursorY
)
```

Parameters

n32CursorX

-

n32CursorY

-

EPointShape.Draw

Draws a graphical representation of a shape, as defined by [EDrawingMode](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)
```

```

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)

```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingMode

Indicates how the point location or model fitting gauge must be displayed, as defined by [EDrawingMode](#).

daughters

TRUE if the daughters gauges are to be displayed also.

color

-

EPointShape.DrawWithCurrentPen

Draws a graphical representation of a shape, as defined by [EDrawingMode](#).

Namespace: Euresys.Open_eVision_2_10

```

[C#]

void DrawWithCurrentPen(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)

```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingMode

Indicates how the point location or model fitting gauge must be displayed, as defined by [EDrawingMode](#).

daughters

TRUE if the daughters gauges are to be displayed also.

EPointShape.HitTest

Checks if there is a handle under the cursor.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool HitTest(
    bool bDaughters
)
```

Parameters

bDaughters

Indicates if the check must be done in the whole hierarchy or just this object.

EPointShape.operator!=

Compares the instance another EPointShape object and returns TRUE if they are not identical.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```



```
bool operator!=(  
    Euresys.Open_eVision_2_10.EPointShape other  
)
```

Parameters

other

EPointShape object to be compared

EPointShape.operator=

Copies all the data from another EPointShape object into the current EPointShape object

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPointShape operator=(  
    Euresys.Open_eVision_2_10.EPointShape other  
)
```

Parameters

other

EPointShape object to be copied

EPointShape.operator==

Compares the instance another EPointShape object and returns TRUE if they are identical.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool operator==(
    Euresys.Open_eVision_2_10.EPointShape other
)
```

Parameters

other

EPointShape object to be compared

EPointShape.SetCenterXY

Sets the center coordinates of a [EPointShape](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetCenterXY(
    float centerX,
    float centerY
)
```

Parameters

centerX

Center coordinates of the [EPointShape](#) object.

centerY

Center coordinates of the [EPointShape](#) object.

EPointShape.Type

Shape type.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
override Euresys.Open_eVision_2_10.EShapeType Type
{ get; }
```

4.149. EPolygonRegion Class

Manages a complete context for an [ERegion](#) shaped like a polygon.

Base Class: [ERegion](#)

Namespace: Euresys.Open_eVision_2_10

Properties

Points

M List of vertices of the region

e

Methods

Drag

Moves the specified handle to a new position and updates all placement parameters of the region.

EPolygonRegion

Constructs an [EPolygonRegion](#) context.

HitTest

Detects if the cursor is placed over one of the dragging handles.

InsertPoint

Insert a vertice between two existing ones

Load	Loads the EPolygonRegion . The given ESerializer must have been created for reading.
operator!=	Checks if this EPolygonRegion instance is not strictly equal to another
operator=	Assignment operator.
operator==	Checks if this EPolygonRegion instance is strictly equal to another
RemovePoint	Remove a vertice
Rotate	Creates a new EPolygonRegion by rotating the EPolygonRegion .
Save	Saves the EPolygonRegion . The given ESerializer must have been created for writing.
Scale	Creates a new EPolygonRegion by scaling the region.
Translate	Creates a new EPolygonRegion by translating the EPolygonRegion .

P

olygonRegion.Drag

Moves the specified handle to a new position and updates all placement parameters of the region.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Drag(
    int x,
    int y,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)
```

Parameters

x

x-coordinate of the mouse cursor.

y

y-coordinate of the mouse cursor.

zoomX

Horizontal zoom factor. By default, true scale is used.

zoomY

Vertical zoom factor. By default, true scale is used.

panX

Horizontal pan offset. By default, no pan is added.

panY

Vertical pan offset. By default, no pan is added.

Remarks

If zooming and/or panning were used when drawing the region, the same values must be used with [EPolygonRegion::HitTest](#) and [EPolygonRegion::Drag](#).

EPolygonRegion.EPolygonRegion

Constructs an [EPolygonRegion](#) context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EPolygonRegion(
)

void EPolygonRegion(
    Euresys.Open_eVision_2_10.EPoint[] points
)

void EPolygonRegion(
    Euresys.Open_eVision_2_10.EPolygonRegion other
)
```

Parameters

points

The list of vertices of the [EPolygonRegion](#).

other

[EPolygonRegion](#) context to copy.

EPolygonRegion.HitTest

Detects if the cursor is placed over one of the dragging handles.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EEditionMode HitTest(
    int x,
    int y,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)
```

Parameters

x

x-coordinate of the mouse cursor.

y

y-coordinate of the mouse cursor.

zoomX

Horizontal zoom factor. By default, true scale is used.

zoomY

Vertical zoom factor. By default, true scale is used.

panX

Horizontal pan offset. By default, no pan is added.

panY

Vertical pan offset. By default, no pan is added.

Remarks

Returns a handle identifier, as defined by [EEditionMode](#).

If zooming and/or panning were used when drawing the region, the same values must be used with [EPolygonRegion::HitTest](#) and [EPolygonRegion::Drag](#).

EPolygonRegion.InsertPoint

Insert a vertice between two existing ones

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool InsertPoint(
    int x,
    int y,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)
```

Parameters

x

x-coordinate of the mouse cursor.

y

y-coordinate of the mouse cursor.

zoomX

Horizontal zoom factor. By default, true scale is used.

zoomY

Vertical zoom factor. By default, true scale is used.

panX

Horizontal pan offset. By default, no pan is added.

panY

Vertical pan offset. By default, no pan is added.

Remarks

If zooming and/or panning were used when drawing the region, the same values must be used with [EPolygonRegion::HitTest](#) and [EPolygonRegion::Drag](#).

EPolygonRegion.Load

Loads the [EPolygonRegion](#). The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Load(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The serializer.

EPolygonRegion.operator!=

Checks if this [EPolygonRegion](#) instance is not strictly equal to another

Namespace: Euresys.Open_eVision_2_10


```
[C#]
bool operator!=(
    Euresys.Open_eVision_2_10.EPolygonRegion other
)
```

Parameters

other

Reference to the other [EPolygonRegion](#) instance

EPolygonRegion.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPolygonRegion operator=(
    Euresys.Open_eVision_2_10.EPolygonRegion other
)
```

Parameters

other

Reference to the [EPolygonRegion](#) used for the assignment

EPolygonRegion.operator==

Checks if this [EPolygonRegion](#) instance is strictly equal to another

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool operator==(
    Euresys.Open_eVision_2_10.EPolygonRegion other
)
```

Parameters

other

Reference to the other [EPolygonRegion](#) instance

EPolygonRegion.Points

List of vertices of the region

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint[] Points
{ get; set; }
```

EPolygonRegion.RemovePoint

Remove a vertice

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
bool RemovePoint(  
    int x,  
    int y,  
    float zoomX,  
    float zoomY,  
    int panX,  
    int panY  
)
```

Parameters

x

x-coordinate of the mouse cursor.

y

y-coordinate of the mouse cursor.

zoomX

Horizontal zoom factor. By default, true scale is used.

zoomY

Vertical zoom factor. By default, true scale is used.

panX

Horizontal pan offset. By default, no pan is added.

panY

Vertical pan offset. By default, no pan is added.

Remarks

If zooming and/or panning were used when drawing the region, the same values must be used with [EPolygonRegion::HitTest](#) and [EPolygonRegion::Drag](#).

EPolygonRegion.Rotate

Creates a new [EPolygonRegion](#) by rotating the [EPolygonRegion](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPolygonRegion Rotate(
    float angle
)
```

Parameters

angle

rotation angle

EPolygonRegion.Save

Saves the [EPolygonRegion](#). The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The serializer.

EPolygonRegion.Scale

Creates a new [EPolygonRegion](#) by scaling the region.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPolygonRegion Scale(
    float scale
)

Euresys.Open_eVision_2_10.EPolygonRegion Scale(
    float scaleX,
    float scaleY
)
```

Parameters

scale

Isotropic scale

scaleX

Horizontal scale

scaleY

Vertical scale

EPolygonRegion.Translate

Creates a new [EPolygonRegion](#) by translating the [EPolygonRegion](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPolygonRegion Translate(
    float dx,
    float dy
)
```

Parameters

dx

Horizontal translation in pixel value

dy

Vertical translation in pixel value

4.150. EPrincipalAxisExtractor Class

A [EPrincipalAxisExtractor](#) object computes the principal axis analysis (PCA) on an [EPointCloud](#) and produces a [E3DTransformMatrix](#) as a result.

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

[ReferenceTransform](#)

M Sets/Gets the reference transform ([E3DTransformMatrix](#)). If not set, it will use the main basis as reference to select the direction of each axis of the new basis.

Methods

[EPrincipalAxisExtractor](#)

Creates an [EPrincipalAxisExtractor](#) object.

[Extract](#)

Computes the [E3DTransformMatrix](#) for a given [EPointCloud](#). This will compute the PCA, then select the direction of each axis of the basis so that this basis is as close as possible as the reference transform. Optionally returns the standard deviation along the 3 axis.

[HasReferenceTransformSet](#)

Returns 'true' if a reference transform ([E3DTransformMatrix](#)) has been set explicitly.

[Load](#)

Loads the [EPrincipalAxisExtractor](#) object configuration. The given [ESerializer](#) must have been created for reading.

operator=

Assignment operator.

Save

Saves the [EPrincipalAxisExtractor](#) object configuration. The given [ESerializer](#) must have been created for writing.

UnsetReferenceTransform

Unset the reference transform ([E3DTransformMatrix](#)).

P

PrincipalAxisExtractor.EPrincipalAxisExtractor

Creates an [EPrincipalAxisExtractor](#) object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void EPrincipalAxisExtractor (
)
void EPrincipalAxisExtractor (
    Euresys.Open_eVision_2_10.Easy3D.EPrincipalAxisExtractor other
)
```

Parameters

other

The object used for the initialization

EPrincipalAxisExtractor.Extract

Computes the [E3DTransformMatrix](#) for a given [EPointCloud](#).

This will compute the PCA, then select the direction of each axis of the basis so that this basis is as close as possible as the reference transform.

Optionally returns the standard deviation along the 3 axis.

Namespace: Euresys.Open_eVision_2_10.Easy3D

[C#]

```
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix Extract(  
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud pc  
)  
  
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix Extract(  
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud pc,  
    ref float stdDevX,  
    ref float stdDevY,  
    ref float stdDevZ  
)
```

Parameters

pc

Input point cloud.

stdDevX

Variable to store the X component of the standard deviation.

stdDevY

Variable to store the Y component of the standard deviation.

stdDevZ

Variable to store the Z component of the standard deviation.

EPrincipalAxisExtractor.HasReferenceTransformSet

Returns 'true' if a reference transform ([E3DTransformMatrix](#)) has been set explicitly.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
bool HasReferenceTransformSet(  
)
```

EPrincipalAxisExtractor.Load

Loads the [EPrincipalAxisExtractor](#) object configuration. The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void Load(  
    Euresys.Open_eVision_2_10.ESerializer serializer  
)
```

Parameters

serializer

The [ESerializer](#) object that is read from.

EPrincipalAxisExtractor.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EPrincipalAxisExtractor operator=(
    Euresys.Open_eVision_2_10.Easy3D.EPrincipalAxisExtractor other
)
```

Parameters

other

The [EPrincipalAxisExtractor](#) object that should be copied.

EPrincipalAxisExtractor.ReferenceTransform

Sets/Gets the reference transform ([E3DTransformMatrix](#)). If not set, it will use the main basis as reference to select the direction of each axis of the new basis.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix ReferenceTransform
{ get; set; }
```

EPrincipalAxisExtractor.Save

Saves the [EPrincipalAxisExtractor](#) object configuration. The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
void Save(  
    Euresys.Open_eVision_2_10.ESerializer serializer  
)
```

Parameters

serializer

The [ESerializer](#) object that is written to.

EPrincipalAxisExtractor.UnsetReferenceTransform

Unset the reference transform ([E3DTransformMatrix](#)).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void UnsetReferenceTransform(  
)
```

4.151. EPseudoColorLookup Class

Describes a lookup table, that is used to for pseudo-coloring (i.e. for assigning colors to gray-level images).

Namespace: Euresys.Open_eVision_2_10

Methods

[EPseudoColorLookup](#)

Default constructor of EPseudoColorLookup objects.

SetShading

Sets up a pseudo-color mapping such that gray level **0** corresponds to color **c24Black**, gray level **255** corresponds to color **c24White**, and intermediate values are interpolated linearly between these two extremes.

pseudoColorLookup.EPseudoColorLookup

Default constructor of EPseudoColorLookup objects.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EPseudoColorLookup(
    Euresys.Open_eVision_2_10.EPseudoColorLookup other
)
void EPseudoColorLookup(
)
```

Parameters

other

-

EPseudoColorLookup.SetShading

Sets up a pseudo-color mapping such that gray level **0** corresponds to color **c24Black**, gray level **255** corresponds to color **c24White**, and intermediate values are interpolated linearly between these two extremes.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetShading(
    Euresys.Open_eVision_2_10.EC24 black,
    Euresys.Open_eVision_2_10.EC24 white,
    Euresys.Open_eVision_2_10.EColorSystem colorSystem,
    bool wrap
)
```

Parameters

black

Color to be mapped on a black (value **0**) pixel.

white

Color to be mapped on a white (value **255**) pixel.

colorSystem

Color system in which interpolation takes place.

wrap

If the color system supports a hue component, indicates whether hue wrap around must be applied.

Remarks

Furthermore, interpolation is performed in the designated color system. Even though interpolation is performed in an arbitrary color system, the extreme colors are specified in the RGB space. To obtain interesting shades of colors, it is recommended to interpolate on the hue component alone.

4.152. EQRCODE Class

Represents a QR code found in the search field.

Namespace: Euresys.Open_eVision_2_10

Properties

DecodedStream

Decoded stream.

Geometry

Geometry of the QR code.

IsDecodingReliable

Decoding reliability.

Level

Level of the QR code.

Model

Model of the QR code.

UnusedErrorCorrection

Unused error correction.

Version

MVersion of the QR code.

e

thods

Draw

Draws the QR code using a pre-defined pen.

DrawWithCurrentPen

Draws the QR code using the pen currently set in the graphical context.

EQRCODE

Creates an EQRCODE object.

operator=

-

EQRCODE.DecodedStream

Decoded stream.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EQRCODE.DecodedStream DecodedStream
{ get; }
```

EQRCODE.Draw

Draws the QR code using a pre-defined pen.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Draw(
    IntPtr hDC,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

Parameters

hDC

-

zoomX

-

zoomY

-
panX
-
panY
-

QRCode.DrawWithCurrentPen

Draws the QR code using the pen currently set in the graphical context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void DrawWithCurrentPen(  
    IntPtr hDC,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

Parameters

hDC
-
zoomX
-
zoomY
-
panX
-
panY
-

EQRCODE.EQRCODE

Creates an EQRCODE object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EQRCODE (
)
void EQRCODE (
    Euresys.Open_eVision_2_10.EQRCODE other
)
```

Parameters

other

-

EQRCODE.Geometry

Geometry of the QR code.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EQRCODEGeometry Geometry
{ get; }
```

EQRCode.IsDecodingReliable

Decoding reliability.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool IsDecodingReliable  
    { get; }
```

EQRCode.Level

Level of the QR code.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EQRCodeLevel Level  
    { get; }
```

EQRCode.Model

Model of the QR code.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EQRCodeModel Model  
  
{ get; }
```

EQRCode.operator=

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EQRCode operator=(  
    Euresys.Open_eVision_2_10.EQRCode other  
)
```

Parameters

other

-

EQRCode.UnusedErrorCorrection

Unused error correction.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float UnusedErrorCorrection
{ get; }
```

Remarks

Returns the amount of unused error correction as a percentage. This parameter ranges from 0 to 1. Returns -1 if error correction failed (too many errors).

EQRCODE.Version

Version of the QR code.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint Version
{ get; }
```

4.153. EQRCODEDecodedStream Class

Represents the complete decoded stream extracted from a QR code.

Namespace: Euresys.Open_eVision_2_10

Properties

[ApplicationIndicator](#)

Application indicator.

CodingMode

Coding mode.

DecodedStreamParts

Decoded stream parts.

RawBitstream

M Raw bit stream.

e

thods

EQRCodeDecodedStream

Creates an EQRCodeDecodedStream object.

operator=

E-

Q

RCodeDecodedStream.ApplicationIndicator

Application indicator.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint ApplicationIndicator
```

```
{ get; }
```

Remarks

The application indicator is relevant if the coding mode of the QR code is FNC1/AIM only.

QRCodeDecodedStream.CodingMode

Coding mode.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.QRCodeCodingMode CodingMode  
{ get; }
```

QRCodeDecodedStream.DecodedStreamParts

Decoded stream parts.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.QRCodeDecodedStreamPart[] DecodedStreamParts  
{ get; }
```

QRCodeDecodedStream.QRCodeDecodedStream

Creates an QRCodeDecodedStream object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EQRCodedDecodedStream(
)
void EQRCodedDecodedStream(
    Euresys.Open_eVision_2_10.EQRCodedDecodedStream other
)
```

Parameters

other

-

EQRCodedDecodedStream.operator=

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EQRCodedDecodedStream operator=(
    Euresys.Open_eVision_2_10.EQRCodedDecodedStream other
)
```

Parameters

other

-

EQRCodedDecodedStream.RawBitstream

Raw bit stream.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
byte[] RawBitstream
```

```
{ get; }
```

Remarks

The raw bit stream is the bit stream of the QR code after unmasking and error correction, but before decoding.

4.154. EQRCodedDecodedStreamPart Class

Represents part of a decoded stream extracted from a QR code.

Namespace: Euresys.Open_eVision_2_10

Properties

[DecodedData](#)

Decoded data.

[Encoding](#)

M Encoding.

e

Methods

[EQRCodedDecodedStreamPart](#)

Creates an [EQRCodedDecodedStreamPart](#) object.

[operator=](#)

-

QRCodeDecodedStreamPart.DecodedData

Decoded data.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
byte [] DecodedData  
    { get; }
```

QRCodeDecodedStreamPart.Encoding

Encoding.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.QRCodeEncoding Encoding  
    { get; }
```

QRCodeDecodedStreamPart.QRCodeDecodedStreamPart

Creates an [QRCodeDecodedStreamPart](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EQRCodedDecodedStreamPart(
)
void EQRCodedDecodedStreamPart(
    Euresys.Open_eVision_2_10.EQRCodedDecodedStreamPart other
)
```

Parameters

other

-

EQRCodedDecodedStreamPart.operator=

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EQRCodedDecodedStreamPart operator=(
    Euresys.Open_eVision_2_10.EQRCodedDecodedStreamPart other
)
```

Parameters

other

-

4.155. EQRCodedGeometry Class

Represents the geometry of a QR code.

Namespace: Euresys.Open_eVision_2_10

Properties

FinderPatternCenters

Finder patterns centers.

Position

M Position of the QR code.

e

Methods

Draw

Draws the QR code geometry using a pre-defined pen.

DrawWithCurrentPen

Draws the QR code geometry using the pen currently set in the graphical context.

EQRCodeGeometry

Creates an [EQRCodeGeometry](#) object.

operator=

E-

Q

RCodeGeometry.Draw

Draws the QR code geometry using a pre-defined pen.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void Draw(  
    IntPtr hDC,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

Parameters

hDC

-

zoomX

-

zoomY

-

panX

-

panY

-

QRCodeGeometry.DrawWithCurrentPen

Draws the QR code geometry using the pen currently set in the graphical context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void DrawWithCurrentPen(  
    IntPtr hDC,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

Parameters

hDC

-

zoomX

-

zoomY

-

panX

-

panY

-

EQRCODEGEOMETRY.EQRCODEGEOMETRY

Creates an [EQRCODEGEOMETRY](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void EQRCODEGEOMETRY(  
)  
  
void EQRCODEGEOMETRY(  
    Euresys.Open_eVision_2_10.EQRCODEGEOMETRY other  
)
```

```
void EQRCodeGeometry(  
    Euresys.Open_eVision_2_10.EQuadrilateral position,  
    Euresys.Open_eVision_2_10.EPoint[] finderPatternCenters  
)
```

Parameters

other

-

position

-

finderPatternCenters

-

EQRCodeGeometry.FinderPatternCenters

Finder patterns centers.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint[] FinderPatternCenters  
{ get; }
```

Remarks

In case of a Micro QR code, there is only one finder pattern center. In case of another QR code, there are three finder pattern centers, returned in the following order: bottom left, top left, top right.

EQRCodeGeometry.operator=

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EQRCodeGeometry operator=(
    Euresys.Open_eVision_2_10.EQRCodeGeometry other
)
```

Parameters

other

-

EQRCodeGeometry.Position

Position of the QR code.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EQuadrilateral Position
{ get; }
```

4.156. EQRCodeReader Class

Represents the QR code reader, that is a context for the detection and decoding of QR codes.

Namespace: Euresys.Open_eVision_2_10

Properties

CellPolarityConfidenceThreshold

Sets the minimum cell polarity confidence threshold. When the cell confidence is under the threshold, additional processing is attempted to improve the polarity detection.

DetectionMethod

Sets the detection method for finding QR codes. The method can be any combination of the `EQRDetectionMethod` enums.

DetectionTradeOff

This setting controls the trade-off between computation speed versus reliability of the detection methods. Setting the trade-off will overwrite the current settings for `EQRCODESCANPRECISION` and `EQRDetectionMethod`.

FilterOutUnreliablyDecodedQRcodes

Activate or deactivate the filtering of unreliably decoded QR codes.

ForegroundDetectionThreshold

Foreground detection threshold. This parameter determines how many grayscale-values a pixel should deviate from its local background to be considered part of the foreground.

MaximumVersion

Maximum version of QR codes to be searched for.

MinimumIsotropy

QR code minimum isotropy.

MinimumScore

Minimum pattern finder score that must be reached to consider that a finder pattern has been found.

MinimumVersion

Minimum version of QR codes to be searched for.

PerspectiveMode

Sets the perspective mode.

ScanPrecision

Precision of the QR code reader when scanning the search field.

SearchedModels

QR code models to be searched for.

SearchField

Search field for the QR code reader.

TimeOut

M Time-out for the [EQRCoderReader::Detect](#), [EQRCoderReader::Decode](#) and [EQRCoderReader::Read](#) methods.

e

thods

Decode

Decodes a QR code candidate, represented as a geometry.

Detect

Detects all QR code candidates in the search field, and returns them as a vector of geometries.

EQRCoderReader

Creates an EQRCoderReader object.

Read

Detects and decodes all the QR codes in the search field.

EQRCoderReader.CellPolarityConfidenceThreshold

Sets the minimum cell polarity confidence threshold. When the cell confidence is under the threshold, additional processing is attempted to improve the polarity detection.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float CellPolarityConfidenceThreshold  
  
    { get; set; }
```

EQRCoderReader.Decode

Decodes a QR code candidate, represented as a geometry.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EQRCoder Decode(  
    Euresys.Open_eVision_2_10.EQRCoderGeometry geometries  
)
```

Parameters

geometries

-

Remarks

The geometry argument can either be custom-built or retrieved after a [EQRCoderReader::Detect](#).

EQRCodeReader.Detect

Detects all QR code candidates in the search field, and returns them as a vector of geometries.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EQRCodeGeometry[] Detect(  
    )
```

Remarks

[EQRCodeReader::Detect](#) only returns candidate QR codes. These candidates can only be confirmed as actual QR codes after a successful decoding.

EQRCodeReader.DetectionMethod

Sets the detection method for finding QR codes. The method can be any combination of the [EQRDetectionMethod](#) enums.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int DetectionMethod  
    { get; set; }
```

Remarks

The default value is: 'EQRDetectionMethod_Gradient|EQRDetectionMethod_AdaptiveThreshold'.

EQRCodeReader.DetectionTradeOff

This setting controls the trade-off between computation speed versus reliability of the detection methods. Setting the trade-off will overwrite the current settings for EQRCodeScanPrecision and EQRDetectionMethod.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EQRDetectionTradeOff DetectionTradeOff
{ get; set; }
```

Remarks

The default value is **EQRCodeDetectionTradeOff_Balanced**.

EQRCodeReader.EQRCodeReader

Creates an EQRCodeReader object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EQRCodeReader (
)
void EQRCodeReader (
    Euresys.Open_eVision_2_10.EQRCodeReader other
)
```

Parameters

other

-

EQRCodeReader.FilterOutUnreliablyDecodedQRCodes

Activate or deactivate the filtering of unreliably decoded QR codes.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool FilterOutUnreliablyDecodedQRCodes  
    { get; set; }
```

Remarks

By default, the QR code reader does not return unreliably decoded QR codes.

EQRCodeReader.ForegroundDetectionThreshold

Foreground detection threshold. This parameter determines how many grayscale-values a pixel should deviate from it's local background to be considered part of the foreground.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int ForegroundDetectionThreshold  
    { get; set; }
```

Remarks

The default value for this parameter is 10.

QRCodeReader.MaximumVersion

Maximum version of QR codes to be searched for.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint MaximumVersion  
  
{ get; set; }
```

Remarks

This parameter value ranges from **1** to **40**. Default value: **40**.

QRCodeReader.MinimumIsotropy

QR code minimum isotropy.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float MinimumIsotropy  
  
{ get; set; }
```

Remarks

The isotropy of a QR code is defined as its short side divided by its long side. This parameter value ranges from 0 to 1. default value: 0.8.

QRCodeReader.MinimumScore

Minimum pattern finder score that must be reached to consider that a finder pattern has been found.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float MinimumScore  
  
{ get; set; }
```

Remarks

The pattern finder score is based on a normalized correlation with a perfect finder pattern model. A perfect match with the model would return a score of 1. This parameter value ranges from **0** to **1**. Default value: **0.65**.

QRCodeReader.MinimumVersion

Minimum version of QR codes to be searched for.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint MinimumVersion  
  
{ get; set; }
```

Remarks

This parameter value ranges from **1** to **40**. Default value: **1**.

EQRCodeReader.PerspectiveMode

Sets the perspective mode.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EQRCodePerspectiveMode PerspectiveMode
{ get; set; }
```

Remarks

The default value is Basic. This setting is deprecated as per Open eVision release 2.0, setting this variable will have no effect. The EQRCodePerspectiveMode_Basic option has been superseded by EQRDetectionMethod_GradientLegacy, the EQRCodePerspectiveMode_Improved option has been superseded by EQRDetectionMethod_PerspectiveLegacy.

EQRCodeReader.Read

Detects and decodes all the QR codes in the search field.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EQRCode[] Read(
)
```

EQRCodeReader.ScanPrecision

Precision of the QR code reader when scanning the search field.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EQRCodeScanPrecision ScanPrecision  
    { get; set; }
```

Remarks

The default value is **EQRCodeScanPrecision_Automatic**.

EQRCodeReader.SearchedModels

QR code models to be searched for.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EQRCodeModel[] SearchedModels  
    { get; set; }
```

Remarks

By default, the QR code reader searches for all models of QR codes.

EQRCodeReader.SearchField

Search field for the QR code reader.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EROIBW8 SearchField
{ get; set; }
```

EQrcodeReader.TimeOut

Time-out for the [EQrcodeReader::Detect](#), [EQrcodeReader::Decode](#) and [EQrcodeReader::Read](#) methods.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
System.UInt64 TimeOut
{ get; set; }
```

Remarks

If the processing time of one of these functions becomes longer than the set time-out, the processing is stopped and an exception is thrown. In that case, the error code of the exception is [TimeoutReached](#). The time-out is set in microseconds. This time-out is not a real time-out. The processing is stopped as soon as possible after the time-out has been reached. This means that the time elapsed effectively in the method can be greater than the time-out in itself.

4.157. EQuadrangle Class

This class represents a polygon with four corners (with sub-pixel accuracy).

Remarks

A quadrangle especially arises when representing the corners of a rotated bounding box.

Namespace: Euresys.Open_eVision_2_10

Properties

GravityCenter

M Returns the gravity center of the quadrangle.

e

Methods

Draw

Draws the quadrangle, by drawing lines between its corners.

DrawWithCurrentPen

Draws the quadrangle, by drawing lines between its corners.

EQuadrangle

Constructs a [EQuadrangle](#). The default constructor initializes all points to 0.

GetPoint

Returns the coordinate of a given corner of the quadrangle.

GetSideAngle

Returns the angle of a given side of the quadrangle.

IsInside

Tests if a point is inside the quadrangle.

operator=

Assignment operator.

OverLaps

Tests if the quadrangles overlap.

SetPoint

Sets the coordinates of a given corner of the quadrangle.

EQuadrangle.Draw

Draws the quadrangle, by drawing lines between its corners.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Draw(
    IntPtr graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    bool drawDiagonals
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    bool drawDiagonals
)

void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    bool drawDiagonals
)
```

Parameters

graphicContext

Graphic context on which to draw.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

drawDiagonals

Specifies whether or not lines are to be drawn between the 1st and 3rd corners, as well as between the 2nd and 4th corners.

color

The color in which to draw the overlay.

Remarks

Drawing is done in the device context associated to the desired window.

EQuadrangle.DrawWithCurrentPen

Draws the quadrangle, by drawing lines between its corners.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawWithCurrentPen(
    IntPtr graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    bool drawDiagonals
)
```

Parameters

graphicContext

Graphic context on which to draw.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

drawDiagonals

Specifies whether or not lines are to be drawn between the 1st and 3rd corners, as well as between the 2nd and 4th corners.

Remarks

Drawing is done in the device context associated to the desired window.

EQuadrangle.EQuadrangle

Constructs a [EQuadrangle](#). The default constructor initializes all points to 0.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EQuadrangle (
)
void EQuadrangle (
    Euresys.Open_eVision_2_10.EQuadrangle other
)
```

Parameters

other

The source [EQuadrangle](#).

EQuadrangle.GetPoint

Returns the coordinate of a given corner of the quadrangle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint GetPoint(  
    uint index  
)
```

Parameters

index

The index of the corner of interest (must lie in the range between 0 and 3, inclusive).

EQuadrangle.GetSideAngle

Returns the angle of a given side of the quadrangle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float GetSideAngle(  
    uint sideIndex  
)
```

Parameters

sideIndex

-

EQuadrangle.GravityCenter

Returns the gravity center of the quadrangle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint GravityCenter
{ get; }
```

EQuadrangle.IsInside

Tests if a point is inside the quadrangle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool IsInside(
    float x,
    float y,
    bool includeEdges
)
```

Parameters

x
-
y
-
includeEdges
-

EQuadrangle.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EQuadrangle operator=(  
    Euresys.Open_eVision_2_10.EQuadrangle other  
)
```

Parameters

other

The source [EQuadrangle](#).

EQuadrangle.OverLaps

Tests if the quadrangles overlap.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool OverLaps(  
    Euresys.Open_eVision_2_10.EQuadrangle other  
)
```

Parameters

other

-

EQuadrangle.SetPoint

Sets the coordinates of a given corner of the quadrangle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetPoint(
    uint index,
    Euresys.Open_eVision_2_10.EPoint location
)
```

Parameters

index

The index of the corner of interest (must lie in the range between 0 and 3, inclusive).

location

The coordinate.

4.158. EQuadrilateral Class

Represents a quadrilateral.

Namespace: Euresys.Open_eVision_2_10

Properties

Corners

The corners of the quadrilateral.

Methods

[EQuadrilateral](#)

Creates an [EQuadrilateral](#) object.

`operator=`

| E

Q

quadrilateral.Corners

The corners of the quadrilateral.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint[] Corners
{ get; }
```

Remarks

The corners are returned in the following order: bottom left, top left, top right, bottom right.

EQuadrilateral.EQuadrilateral

Creates an [EQuadrilateral](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EQUadrilateral(
)

void EQUadrilateral(
    Euresys.Open_eVision_2_10.EPoint[] corners
)

void EQUadrilateral(
    Euresys.Open_eVision_2_10.EQuadrilateral other
)
```

Parameters

corners

-

other

-

EQuadrilateral.operator=

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EQuadrilateral operator=(
    Euresys.Open_eVision_2_10.EQuadrilateral other
)
```

Parameters

other

-

4.159. ERandomDecimator Class

Decimation of an [EPointCloud](#).

The random decimator decimates a point cloud by copying a specified number of points, randomly selected, to a new point cloud.

Base Class: [EDecimator](#)

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

NumberOfPoints

M Sets/gets the parameter "number of points" (number of points after decimation).

e

Methods

Decimate

Decimates a given [EPointCloud](#) and writes the result into a new point cloud.

ERandomDecimator

Creates an [ERandomDecimator](#) object. If the required number of points (after decimation) is not specified, the default value is **10000**.

operator=

Assignment operator.

Serialize

Serializer

ERandomDecimator.Decimate

Decimates a given [EPointCloud](#) and writes the result into a new point cloud.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Decimate (
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud cloudIn,
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud cloudOut
)
```

Parameters

cloudIn

The input point cloud.

cloudOut

The output point cloud.

Remarks

The input point cloud 'cloudIn' should be different from the output point cloud 'cloudOut'. If not an exception will be thrown.

ERandomDecimator.ERandomDecimator

Creates an [ERandomDecimator](#) object. If the required number of points (after decimation) is not specified, the default value is **10000**.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
void ERandomDecimator(  
    )  
  
void ERandomDecimator(  
    int numberOfPoints  
    )  
  
void ERandomDecimator(  
    Euresys.Open_eVision_2_10.Easy3D.ERandomDecimator other  
    )
```

Parameters

numberOfPoints

Number of points after decimation.

other

Reference to the [ERandomDecimator](#) object used for the initialization.

ERandomDecimator.NumberOfPoints

Sets/gets the parameter "number of points" (number of points after decimation).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
int NumberOfPoints  
  
    { get; set; }
```

ERandomDecimator.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.ERandomDecimator operator=(
    Euresys.Open_eVision_2_10.Easy3D.ERandomDecimator other
)
```

Parameters

other

The [ERandomDecimator](#) object that should be copied.

ERandomDecimator.Serialize

Serializer

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Serialize(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is read from or written to.

4.160. ERectangle Class

Represents a model of a rectangle in EasyGauge.

Base Class: [EFrame](#)

Namespace: Euresys.Open_eVision_2_10

Properties

SizeX

X size of the [ERectangle](#)

SizeY

MY size of the [ERectangle](#)

e

Methods

CopyTo

Copies all the data of the current [ERectangle](#) object into another [ERectangle](#) object, and returns it.

[ERectangle](#)

Constructs a [ERectangle](#)

GetCorners

Retrieves the coordinates of each corner of a [ERectangle](#) object.

GetEdges

Retrieves each edge of a [ERectangle](#) object.

GetMidEdges

Retrieves the center coordinates of each edge of a [ERectangle](#) object.

GetPoint

Returns the coordinates of a particular point, specified by its location in the [ERectangle](#) area.

operator=

Copies all the data from another [ERectangle](#) object into the current [ERectangle](#) object

[SetFromOppositeCorners](#)

Sets the geometric parameters (center coordinates, size, and rotation angle) of an [ERectangle](#) object.

[SetFromOriginMiddleEnd](#)

DEPRECATED (you should use [ERectangle::SetFromThreeCorners](#)): Sets the geometric parameters (center coordinates, size, and rotation angle) of an [ERectangle](#) object.

[SetFromThreeCorners](#)

Sets the geometric parameters (center coordinates, size, and rotation angle) of an [ERectangle](#) object.

[SetFromTwoPoints](#)

DEPRECATED (you should use [ERectangle::SetFromOppositeCorners](#)): Sets the geometric parameters (center coordinates, size, and rotation angle) of an [ERectangle](#) object.

[SetSize](#)

⌈ Sets the size of a [ERectangle](#) object.

R

rectangle.CopyTo

Copies all the data of the current [ERectangle](#) object into another [ERectangle](#) object, and returns it.

Namespace: `Euresys.Open_eVision_2_10`

```
[C#]
Euresys.Open_eVision_2_10.ERectangle CopyTo(
    Euresys.Open_eVision_2_10.ERectangle other
)
```

Parameters

other

Pointer to the ERectangle object in which the current ERectangle object data have to be copied.

Remarks

In case of a **NULL** pointer, a new ERectangle object will be created and returned.

ERectangle.ERectangle

Constructs a [ERectangle](#)

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ERectangle(
)
void ERectangle(
    Euresys.Open_eVision_2_10.EPoint center,
    float sizeX,
    float sizeY,
    float angle
)
void ERectangle(
    Euresys.Open_eVision_2_10.EPoint origin,
    Euresys.Open_eVision_2_10.EPoint end
)
void ERectangle(
    Euresys.Open_eVision_2_10.EPoint origin,
    Euresys.Open_eVision_2_10.EPoint middle,
    Euresys.Open_eVision_2_10.EPoint end
)
void ERectangle(
    Euresys.Open_eVision_2_10.ERectangle other
)
```

Parameters

center

Center coordinates of the rectangle at its nominal position. The default value is **(0,0)**.

sizeX

Nominal size X/Y of the rectangle. Both default values are **100**.

sizeY

Nominal size X/Y of the rectangle. Both default values are **100**.

angle

Nominal rotation angle of the rectangle. The default value is **0**.

origin

Upper left point coordinates of the rectangle.

end

Lower right point coordinates of the rectangle.

middle

A third corner point coordinates.

other

Another [ERectangle](#) object to be copied in the new [ERectangle](#) object.

ERectangle.GetCorners

Retrieves the coordinates of each corner of a [ERectangle](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetCorners (
    Euresys.Open_eVision_2_10.EPoint xy,
    Euresys.Open_eVision_2_10.EPoint XXy,
    Euresys.Open_eVision_2_10.EPoint xYY,
    Euresys.Open_eVision_2_10.EPoint XXYY
)
```

Parameters

xy

Coordinates of the lower leftmost corner of the [ERectangle](#) object.

XXy

Coordinates of the lower rightmost corner of the [ERectangle](#) object.

xYY

Coordinates of the upper leftmost corner of the [ERectangle](#) object.

XXYY

Coordinates of the upper rightmost corner of the [ERectangle](#) object.

ERectangle.GetEdges

Retrieves each edge of a [ERectangle](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetEdges (
    Euresys.Open_eVision_2_10.ELine x,
    Euresys.Open_eVision_2_10.ELine XX,
    Euresys.Open_eVision_2_10.ELine y,
    Euresys.Open_eVision_2_10.ELine YY
)
```

Parameters

x

Leftmost edge of the [ERectangle](#) object.

XX

Rightmost edge of the [ERectangle](#) object.

y

Lower edge of the [ERectangle](#) object.

YY

Upper edge of the [ERectangle](#) object.

ERectangle.GetMidEdges

Retrieves the center coordinates of each edge of a [ERectangle](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetMidEdges (
    Euresys.Open_eVision_2_10.EPoint x,
    Euresys.Open_eVision_2_10.EPoint XX,
    Euresys.Open_eVision_2_10.EPoint y,
    Euresys.Open_eVision_2_10.EPoint YY
)
```

Parameters

x

Center coordinates of the leftmost edge of the [ERectangle](#) object.

XX

Center coordinates of the rightmost edge of the [ERectangle](#) object.

y

Center coordinates of the lower edge of the [ERectangle](#) object.

YY

Center coordinates of the upper edge of the [ERectangle](#) object.

ERectangle.GetPoint

Returns the coordinates of a particular point, specified by its location in the [ERectangle](#) area.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint GetPoint(
    float fractionX,
    float fractionY
)
```

Parameters

fractionX

Point location expressed as a fraction of the [ERectangle](#) vertical edges (range -1, +1).

fractionY

Point location expressed as a fraction of the [ERectangle](#) horizontal edges (range -1, +1).

ERectangle.operator=

Copies all the data from another [ERectangle](#) object into the current [ERectangle](#) object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ERectangle operator=(
    Euresys.Open_eVision_2_10.ERectangle other
)
```

Parameters

other

[ERectangle](#) object to be copied

ERectangle.SetFromOppositeCorners

Sets the geometric parameters (center coordinates, size, and rotation angle) of an [ERectangle](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetFromOppositeCorners(
    Euresys.Open_eVision_2_10.EPoint origin,
    Euresys.Open_eVision_2_10.EPoint end
)
```

Parameters

origin

Upper left point coordinates of the rectangle.

end

Lower right point coordinates of the rectangle.

ERectangle.SetFromOriginMiddleEnd

DEPRECATED (you should use [ERectangle::SetFromThreeCorners](#)): Sets the geometric parameters (center coordinates, size, and rotation angle) of an [ERectangle](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetFromOriginMiddleEnd(
    Euresys.Open_eVision_2_10.EPoint origin,
    Euresys.Open_eVision_2_10.EPoint middle,
    Euresys.Open_eVision_2_10.EPoint end
)
```

Parameters

origin

Upper left point coordinates of the rectangle.

middle

A third corner point coordinates.

end

Lower right point coordinates of the rectangle.

ERectangle.SetFromThreeCorners

Sets the geometric parameters (center coordinates, size, and rotation angle) of an [ERectangle](#) object.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
void SetFromThreeCorners (
    Euresys.Open_eVision_2_10.EPoint origin,
    Euresys.Open_eVision_2_10.EPoint middle,
    Euresys.Open_eVision_2_10.EPoint end
)
```

Parameters

origin

Upper left point coordinates of the rectangle.

middle

A third corner point coordinates.

end

Lower right point coordinates of the rectangle.

ERectangle.SetFromTwoPoints

DEPRECATED (you should use [ERectangle::SetFromOppositeCorners](#)): Sets the geometric parameters (center coordinates, size, and rotation angle) of an [ERectangle](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetFromTwoPoints (
    Euresys.Open_eVision_2_10.EPoint origin,
    Euresys.Open_eVision_2_10.EPoint end
)
```

Parameters

origin

Upper left point coordinates of the rectangle.

end

Lower right point coordinates of the rectangle.

ERectangle.SetSize

Sets the size of a [ERectangle](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetSize(
    float sizeX,
    float sizeY
)
```

Parameters

sizeX

Nominal size X of the [ERectangle](#) object. Default values is **100**.

sizeY

Nominal size Y of the [ERectangle](#) object. Default values is **100**.

Remarks

A [ERectangle](#) object is fully defined knowing its nominal position (given by the coordinates of its center), its nominal size, its rotation angle and its outline tolerance. By default, the width and height values are **100**, which means 100 pixels when the field of view is not calibrated and 100 "units" in case of a calibrated field of view.

ERectangle.SizeX

X size of the [ERectangle](#)

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float SizeX
```

```
{ get; }
```

ERectangle.SizeY

Y size of the [ERectangle](#)

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float SizeY  
  
{ get; }
```

4.161. ERectangleGauge Class

Manages a rectangle fitting gauge.

Base Class: [ERectangleShape](#)

Namespace: Euresys.Open_eVision_2_10

Properties

[Active](#)

Sets the flag indicating whether the gauge is active or not.

[ActiveEdges](#)

Active edges as defined in [EDragHandle](#).

AverageDistance

Average distance between the sampled points and the fitted model.

FilteringThreshold

Relative filtering threshold, that is the fraction of the average distance between the sampled points and the fitted model above which a point is filtered out.

HVConstraint

Status of the restriction on the orientation of the point location gauge or model fitting sample paths.

InnerFilteringEnabled

Getter method for the **GetInnerFilteringEnabled** property. This property is the flag indicating if the inner sampled point filtering is enabled (**TRUE**).

InnerFilteringThreshold

Sampled point inner filtering threshold.

KnownAngle

Flag indicating whether the rotation angle of the rectangle to be fitted is known or not.

MeasuredRectangle

Information pertaining to the fitted rectangle.

MinAmplitude

Offset added to the **Threshold** when a peak is to be detected.

MinArea

Minimum area value.

NumFilteringPasses

Number of filtering passes for a model fitting operation.

NumSamples	Number of sampled points during the model fitting operation.
NumSamplesx	Number of sampled points found on edge x during the measure operation.
NumSamplesX	Number of sampled points found on edge X during the measure operation.
NumSamplesy	Number of sampled points found on edge y during the measure operation.
NumSamplesY	Number of sampled points found on edge Y during the measure operation.
NumSkipRanges	Number of skip ranges in the gauge after a call to ERect-angleGauge::AddSkipRange .
NumValidSamples	Number of valid sample points remaining after a model fitting operation.
RectangularSamplingArea	Flag indicating whether the sampling area remains a rectangle when rotated, instead of becoming a parallelogram.
SamplingStep	Approximate distance between sampled points during a model fitting operation.
Smoothing	Number of pixels used for the low-pass filtering operation.

Thickness

Number of parallel segments used to extract the data profile.

Threshold

Threshold level used to delimit significant peaks in the data profile.

Tolerance

Searching area half thickness of the rectangle fitting gauge.

TransitionChoice

Transition choice.

TransitionIndex

Index (from 0 on) of the transition to be retained when the transition choice parameter is set to [NthFromBegin](#) or [NthFromEnd](#).

TransitionType

Transition type.

Type

Shape type.

Valid

M Flag indicating if at least one valid transition has been found.

e

thods

AddSkipRange

Adds an item to the set of skip ranges and returns the index of the newly added range.

CopyTo

Copies all the data of the current ERectangleGauge object into another ERectangleGauge object, and returns it.

DisableInnerFiltering

Disables inner sampled point filtering.

Drag

Moves a handle to a new position and updates the position parameters of the gauge.

Draw

Draws a graphical representation of a point location or model fitting gauge, as defined by [EDrawingMode](#).

DrawWithCurrentPen

Draws a graphical representation of a point location or model fitting gauge, as defined by [EDrawingMode](#).

ERectangleGauge

Constructs a rectangle measurement context.

GetMeasuredPoint

Returns the coordinates of a sample point, measured along one of the sample paths of the gauge.

GetMinNumFitSamples

Returns the minimum number of samples required for fitting on each side of the shape.

GetSamplex

Allows to retrieve information on the samples found along the x edge.

GetSampleX

Allows to retrieve information on the samples found along the X edge.

GetSampleY

Allows to retrieve information on the samples found along the y edge.

GetSampleY

Allows to retrieve information on the samples found along the Y edge.

GetSkipRange

Allows to retrieve the start and end values of the skip range corresponding to the given index, if it is valid (i.e. if it has previously been created by a call to the [ERectangleGauge::AddSkipRange](#) method).

HitTest

Checks whether the cursor is positioned over a handle (**TRUE**) or not (**FALSE**).

Measure

Triggers the point location or the model fitting operation.

MeasureSample

Computes the sample points along the sample path whose index in the list is given by the **pathIndex** parameter.

MeasureWithoutFitting

Triggers the point location without rectangle fitting operation.

operator=

Copies all the data from another [ERectangleGauge](#) object into the current [ERectangleGauge](#) object

Plot

Draws the profile that is crossed by one of the sample paths of the gauge, as defined by [EPlotItem](#).

[PlotWithCurrentPen](#)

Draws the profile that is crossed by one of the sample paths of the gauge, as defined by [EPlotItem](#).

[Process](#)

Triggers the process pertaining to a shape or gauge and all the daughter gauges attach to it.

[RemoveAllSkipRanges](#)

Removes all the skip ranges previously created by a call to [ERectangleGauge::AddSkipRange](#).

[RemoveSkipRange](#)

After a call to [ERectangleGauge::AddSkipRange](#), removes the skip range with the given index.

[SetMinNumFitSamples](#)

⌊ Sets the minimum number of samples required for fitting on each side of the shape.

R

rectangleGauge.Active

Sets the flag indicating whether the gauge is active or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
override bool Active
    { get; set; }
```

Remarks

When complex gauging is required, several gauges can be grouped together. Applying [ERectangleGauge::Process](#) to the mother gauge or shape triggers the measurement of the whole. Only the active gauges will participate in the process. By default, the gauge is active (**TRUE**).

ERectangleGauge.ActiveEdges

Active edges as defined in [EDragHandle](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint ActiveEdges
    { get; set; }
```

Remarks

In the case of a rectangle fitting gauge, each edge can have its own transition detection parameters. Updating the transition parameters only affect the current active edges. By default, all edges are active.

ERectangleGauge.AddSkipRange

Adds an item to the set of skip ranges and returns the index of the newly added range.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint AddSkipRange(
    uint start,
    uint end
)
```

Parameters

start

Beginning of the skip range.

end

End of the skip range.

Remarks

The samples indices between start and end (including the boundaries) will be discarded during the measurement process.

The [ERectangleGauge::AddSkipRange](#) method allows to define skip ranges in an [ERectangleGauge](#). This means that, at measure time, samples belonging to these ranges will not be taken into account.

A sample may belong to more than one skip range; to be discarded, a sample has to pertain to at least one range.

Moreover, the skip ranges are allowed to overlap one another.

The range is allowed to be reversed (i.e. end is not required to be greater than start).

Also, start and end are not required to reference valid indices at the time of the call (i.e. the range may lie outside of the current return value for [ERectangleGauge::NumSamples](#)).

ERectangleGauge.AverageDistance

Average distance between the sampled points and the fitted model.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float AverageDistance  
{ get; }
```

Remarks

Irrelevant in case of a point location operation.

ERectangleGauge.CopyTo

Copies all the data of the current [ERectangleGauge](#) object into another [ERectangleGauge](#) object, and returns it.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ERectangleGauge CopyTo(
    Euresys.Open_eVision_2_10.ERectangleGauge other,
    bool recursive
)
```

Parameters

other

Pointer to the ERectangleGauge object in which the current ERectangleGauge object data have to be copied.

recursive

TRUE if the children gauges have to be copied as well, **FALSE** otherwise.

Remarks

In case of a **NULL** pointer, a new ERectangleGauge object will be created and returned.

ERectangleGauge.DisableInnerFiltering

Disables inner sampled point filtering.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DisableInnerFiltering(
)
```

Remarks

The inner sampled point filtering is activated as soon as the corresponding [ERectangleGauge::InnerFilteringThreshold](#) is set.

ERectangleGauge.Drag

Moves a handle to a new position and updates the position parameters of the gauge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Drag(
    int x,
    int y
)
```

Parameters

x
Cursor current X coordinate.

y
Cursor current Y coordinate.

ERectangleGauge.Draw

Draws a graphical representation of a point location or model fitting gauge, as defined by [EDrawingMode](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)
```

```

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)

```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingMode

Indicates how the point location or model fitting gauge must be displayed, as defined by [EDrawingMode](#).

daughters

TRUE if the daughters gauges are to be displayed also.

color

The color in which to draw the overlay.

ERectangleGauge.DrawWithCurrentPen

Draws a graphical representation of a point location or model fitting gauge, as defined by [EDrawingMode](#).

Namespace: Euresys.Open_eVision_2_10

```

[C#]

void DrawWithCurrentPen(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)

```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingMode

Indicates how the point location or model fitting gauge must be displayed, as defined by [EDrawingMode](#).

daughters

TRUE if the daughters gauges are to be displayed also.

ERectangleGauge.ERectangleGauge

Constructs a rectangle measurement context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ERectangleGauge (
)
void ERectangleGauge (
    Euresys.Open_eVision_2_10.ERectangleGauge other
)
```

Parameters

other

Another ERectangleGauge object to be copied in the new ERectangleGauge object.

Remarks

With the default constructor, all the parameters are initialized to their respective default values.

With the copy constructor, the constructed rectangle measurement context is based on a pre-existing ERectangleGauge object. The gauge children are also copied. Hierarchy copying through a copy constructor is always recursive. To disable this recursion, use instead the [ERectangleGauge::CopyTo](#) method.

ERectangleGauge.FilteringThreshold

Relative filtering threshold, that is the fraction of the average distance between the sampled points and the fitted model above which a point is filtered out.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float FilteringThreshold  
  
    { get; set; }
```

Remarks

Irrelevant in case of a point location operation.

During a model fitting operation, the "filtering" process can be invoked to remove outliers, i.e. points that were located significantly far away from the fitted model (so that their position is dubious).

ERectangleGauge.GetMeasuredPoint

Returns the coordinates of a sample point, measured along one of the sample paths of the gauge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint GetMeasuredPoint(  
    uint index  
)
```

Parameters

index

This argument must be left unchanged from its default value, i.e. **~0** (= **0xFFFFFFFF**).

Remarks

These coordinates pertain to the world space; they are expressed in the reference frame to which the current `ERectangleGauge` object belongs. The gauging process uses a list of sample points to find the shape position and size that best fit a given image. These sample points are measured along the sample paths defined by the gauge geometry. `ERectangleGauge::GetMeasuredPoint` returns the coordinates of the sample point that meets the following two requirements: 1. It lies on the sample path inspected with the last call to `ERectangleGauge::MeasureSample`, and 1. Among all the sample points along the latter sample path, it is the one selected by the `ERectangleGauge::TransitionChoice` property.

Note. For this method to succeed, it is necessary to previously call `ERectangleGauge::MeasureSample`.

ERectangleGauge.GetMinNumFitSamples

Returns the minimum number of samples required for fitting on each side of the shape.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetMinNumFitSamples (
    out int side0,
    out int side1,
    out int side2,
    out int side3
)
```

Parameters

side0

Minimum number of samples on the top side of the rectangle.

side1

Minimum number of samples on the left side of the rectangle.

side2

Minimum number of samples on the bottom side of the rectangle.

side3

Minimum number of samples on the right side of the rectangle.

Remarks

Irrelevant in case of a point location operation.

ERectangleGauge.GetSampleX

Allows to retrieve information on the samples found along the X edge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool GetSampleX(
    Euresys.Open_eVision_2_10.EPoint pt,
    uint index
)

void GetSampleX(
    Euresys.Open_eVision_2_10.ESamplePoint sp,
    uint index
)

bool GetSampleX(
    ref Euresys.Open_eVision_2_10.EPeak pk,
    uint index
)
```

Parameters

pt

[EPoint](#) structure that will receive the sample position.

index

The sample index

sp

[ESamplePoint](#) structure that will receive the sample position.

pk

[EPeak](#) structure that will contain the sample peak properties.

Remarks

The method provides the sample point corresponding to the supplied index. The returned value is true when the sample is valid, and false otherwise.

ERectangleGauge.GetSampleX

Allows to retrieve information on the samples found along the X edge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool GetSampleX(
    Euresys.Open_eVision_2_10.EPoint pt,
    uint index
)

void GetSampleX(
    Euresys.Open_eVision_2_10.ESamplePoint sp,
    uint index
)

bool GetSampleX(
    ref Euresys.Open_eVision_2_10.EPeak pk,
    uint index
)
```

Parameters

pt

[EPoint](#) structure that will receive the sample position.

index

The sample index

sp

[ESamplePoint](#) structure that will receive the sample position.

pk

[EPeak](#) structure that will contain the sample peak properties.

Remarks

The method provides the sample point corresponding to the supplied index. The returned value is true when the sample is valid, and false otherwise.

ERectangleGauge.GetSampleY

Allows to retrieve information on the samples found along the Y edge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool GetSampleY(
    Euresys.Open_eVision_2_10.EPoint pt,
    uint index
)

void GetSampleY(
    Euresys.Open_eVision_2_10.ESamplePoint sp,
    uint index
)

bool GetSampleY(
    ref Euresys.Open_eVision_2_10.EPeak pk,
    uint index
)
```

Parameters

pt

[EPoint](#) structure that will receive the sample position.

index

The sample index

sp

[ESamplePoint](#) structure that will receive the sample position.

pk

[EPeak](#) structure that will contain the sample peak properties.

Remarks

The method provides the sample point corresponding to the supplied index. The returned value is true when the sample is valid, and false otherwise.

ERectangleGauge.GetSampleY

Allows to retrieve information on the samples found along the Y edge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool GetSampleY(
    Euresys.Open_eVision_2_10.EPoint pt,
    uint index
)

void GetSampleY(
    Euresys.Open_eVision_2_10.ESamplePoint sp,
    uint index
)

bool GetSampleY(
    ref Euresys.Open_eVision_2_10.EPeak pk,
    uint index
)
```

Parameters

pt

[EPoint](#) structure that will receive the sample position.

index

The sample index

sp

[ESamplePoint](#) structure that will receive the sample position.

pk

[EPeak](#) structure that will contain the sample peak properties.

Remarks

The method provides the sample point corresponding to the supplied index. The returned value is true when the sample is valid, and false otherwise.

ERectangleGauge.GetSkipRange

Allows to retrieve the start and end values of the skip range corresponding to the given index, if it is valid (i.e. if it has previously been created by a call to the [ERectangleGauge::AddSkipRange](#) method).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetSkipRange(
    uint index,
    out uint start,
    out uint end
)
```

Parameters

index

Index of the skip range.

start

Beginning of the skip range.

end

End of the skip range.

Remarks

Start is guaranteed to be smaller or equal to end.

ERectangleGauge.HitTest

Checks whether the cursor is positioned over a handle (**TRUE**) or not (**FALSE**).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool HitTest(
    bool daughters
)
```

Parameters

daughters

TRUE if the daughters gauges handles have to be considered as well.

ERectangleGauge.HVConstraint

Status of the restriction on the orientation of the point location gauge or model fitting sample paths.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool HVConstraint
{ get; set; }
```

Remarks

Sample paths are the point location gauges placed along the model to be fitted.

ERectangleGauge.InnerFilteringEnabled

Getter method for the **GetInnerFilteringEnabled** property. This property is the flag indicating if the inner sampled point filtering is enabled (**TRUE**).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool InnerFilteringEnabled
{ get; }
```

Remarks

The inner sampled point filtering is activated as soon as the corresponding threshold is set, getting the [ERectangleGauge.InnerFilteringThreshold](#) property. To disable inner filtering, use the **DisableInnerFiltering** method.

ERectangleGauge.InnerFilteringThreshold

Sampled point inner filtering threshold.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float InnerFilteringThreshold
{ get; set; }
```

Remarks

If inner filtering is enabled, the sampled points that have been found inside the measured rectangle are filtered in regard of their distance to it. If this distance is greater than the threshold, the sampled point is set as invalid, and removed from the measure. This distance is in physical units.

The inner sampled point filtering is activated as soon as the corresponding threshold is set. To disable inner filtering, use the **DisableInnerFiltering** method.

ERectangleGauge.KnownAngle

Flag indicating whether the rotation angle of the rectangle to be fitted is known or not.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
```

```
bool KnownAngle
```

```
{ get; set; }
```

Remarks

A rectangle model to be fitted may have a well-known orientation. It is possible to impose the value of this rotation angle, thus removing one degree of freedom. The rectangle fitting gauge orientation is set by means of the [ERectangleGauge.Angle](#) property.

The sign of the rotation angle depends whether the field of view is calibrated or not.

* When the field of view is calibrated, the coordinate system is said to be direct, the abscissa extends rightwards and the ordinate extends upwards. In this case, an anticlockwise rotation leads to a positive angle value.

* When the field of view is not calibrated, the coordinate system is said to be inverse, the abscissa extends rightwards and the ordinate extends downwards. In this case, a clockwise rotation leads to a positive angle value.

ERectangleGauge.Measure

Triggers the point location or the model fitting operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void Measure(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage  
)
```

Parameters

sourceImage

Pointer to the source image.

Remarks

When this method is called, and if not enough valid points are detected, then the method returns directly, and the measured gauge is set to the nominal parameters.

ERectangleGauge.MeasuredRectangle

Information pertaining to the fitted rectangle.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ERectangle MeasuredRectangle
{ get; }
```

ERectangleGauge.MeasureSample

Computes the sample points along the sample path whose index in the list is given by the **pathIndex** parameter.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void MeasureSample (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    uint pathIndex
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

pathIndex

Sample path index.

Remarks

This method stores its results into a temporary variable inside the ERectangleGauge object.

ERectangleGauge.MeasureWithoutFitting

Triggers the point location without rectangle fitting operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void MeasureWithoutFitting(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage
)
```

Parameters

sourceImage
Source image.

Remarks

This method performs the actual measurement for each transition, but does not perform the rectangle fitting. This means that individual samples will be available for each edges through the [ERectangleGauge::GetSampleX](#) (Edge x), [ERectangleGauge::GetSampleY](#) (Edge y), [ERectangleGauge::GetSampleX](#) (Edge X), [ERectangleGauge::GetSampleY](#) (Edge Y) methods, but the gauge position will not be changed.

Please note that the filtering will not be performed in this method, since it relies upon the fitting process. The filtering parameters will thus be unused.

ERectangleGauge.MinAmplitude

Offset added to the **Threshold** when a peak is to be detected.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint MinAmplitude
```

```
{ get; set; }
```

Remarks

When analyzing a derivative profile, a peak is made up of consecutive pixel values above **Threshold**. To detect weak [strong] transitions, lower [raise] the **Threshold** value.

To avoid interference of noise, an additional parameter is provided. The **MinAmplitude** parameter is an offset added to **Threshold** when a peak is to be detected.

When the pixel values of the derivative profile do not reach **Threshold + MinAmplitude**, the peak is not taken into account. Anyway, when a peak is taken into account, all the pixels with values above **Threshold** are considered (for more accuracy). Setting the **MinAmplitude** value to **0** merely cancels its effect.

ERectangleGauge.MinArea

Minimum area value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
uint MinArea  
  
{ get; set; }
```

Remarks

A transition is detected if its derivative peak reaches **Threshold + MinAmplitude** value, and then declared valid if the area between the peak curve and the horizontal at level **Threshold** reaches the **MinArea** value.

ERectangleGauge.NumFilteringPasses

Number of filtering passes for a model fitting operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint NumFilteringPasses  
  
{ get; set; }
```

Remarks

Irrelevant in case of a point location operation.

During a filtering pass, the points that are too distant from the model are discarded.

During a model fitting operation, the "filtering" process can be invoked to remove outliers, i.e. points that were located significantly far away from the fitted model (so that their position is dubious).

By default (the number of filtering passes is 0), the outliers rejection process is disabled.

ERectangleGauge.NumSamples

Number of sampled points during the model fitting operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint NumSamples  
  
{ get; }
```

Remarks

Irrelevant in case of a point location operation.

After a model fitting operation, a number of points have been fitted along the model. Among them, some are not reliable because of their **Area** value. Among the remaining ones, some are filtered out (**NumFilteringPasses**, **FilteringThreshold**).

ERectangleGauge.NumSamplesX

Number of sampled points found on edge X during the measure operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint NumSamplesX  
    { get; }
```

ERectangleGauge.NumSamplesX

Number of sampled points found on edge X during the measure operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint NumSamplesX  
    { get; }
```

ERectangleGauge.NumSamplesY

Number of sampled points found on edge Y during the measure operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint NumSamplesY  
    { get; }
```

ERectangleGauge.NumSamplesY

Number of sampled points found on edge Y during the measure operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint NumSamplesY  
    { get; }
```

ERectangleGauge.NumSkipRanges

Number of skip ranges in the gauge after a call to [ERectangleGauge::AddSkipRange](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint NumSkipRanges  
    { get; }
```

ERectangleGauge.NumValidSamples

Number of valid sample points remaining after a model fitting operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint NumValidSamples
{ get; }
```

Remarks

Irrelevant in case of a point location operation.

After a model fitting operation, a number of points have been fitted along the model. Among them, some are not reliable because of their **Area** value. Among the remaining ones, some are filtered out (**NumFilteringPasses**, **FilteringThreshold**).

ERectangleGauge.operator=

Copies all the data from another ERectangleGauge object into the current ERectangleGauge object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ERectangleGauge operator=(
    Euresys.Open_eVision_2_10.ERectangleGauge other
)
```

Parameters

other
ERectangleGauge object to be copied

ERectangleGauge.Plot

Draws the profile that is crossed by one of the sample paths of the gauge, as defined by [EPlotItem](#).

Namespace: Euresys.Open_eVision_2_10


```

[C#]
void Plot(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.EPlotItem drawItems,
    float width,
    float height,
    float originX,
    float originY
)

void Plot(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EPlotItem drawItems,
    float width,
    float height,
    float originX,
    float originY
)

void Plot(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.EPlotItem drawItems,
    float width,
    float height,
    float originX,
    float originY
)

```

Parameters

graphicContext

Handle of the device context on which to draw.

drawItems

Boolean combination of [EPlotItem](#) members, that indicates which items must be displayed.

width

Width of the plot.

height

Height of the plot.

originX

Origin point coordinates of the plot along the X axis.

originY

Origin point coordinates of the plot along the Y axis.

color

The color in which to draw the overlay.

Remarks

The sample path that is taken into considered is the one inspected with the last call to [ERectangleGauge::MeasureSample](#).

Note. For this method to succeed, it is necessary to previously call [ERectangleGauge::MeasureSample](#).

ERectangleGauge.PlotWithCurrentPen

Draws the profile that is crossed by one of the sample paths of the gauge, as defined by [EPlotItem](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void PlotWithCurrentPen(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EPlotItem drawItems,
    float width,
    float height,
    float originX,
    float originY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawItems

Boolean combination of [EPlotItem](#) members, that indicates which items must be displayed.

width

Width of the plot.

height

Height of the plot.

originX

Origin point coordinates of the plot along the X axis.

originY

Origin point coordinates of the plot along the Y axis.

Remarks

The sample path that is taken into considered is the one inspected with the last call to [ERectangleGauge::MeasureSample](#).

Note. For this method to succeed, it is necessary to previously call [ERectangleGauge::MeasureSample](#).

ERectangleGauge.Process

Triggers the process pertaining to a shape or gauge and all the daughter gauges attach to it.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Process (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    bool daughters
)
```

Parameters

sourceImage

Pointer to the source image.

daughters

Flag indicating whether the daughters shapes inherit of the same behavior.

Remarks

When complex gauging is required, several gauges can be grouped together. Applying **Process** to the mother gauge or shape triggers the measurement of the whole. Only the active gauges will participate in the process.

ERectangleGauge.RectangularSamplingArea

Flag indicating whether the sampling area remains a rectangle when rotated, instead of becoming a parallelogram.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
bool RectangularSamplingArea  
  
    { get; set; }
```

Remarks

By default, this flag is set to **TRUE**: the sampling area always remains a rectangle. Setting this property is only useful when the thickness transition parameter is greater than 1. In fact, when thickness transition parameter is equal to 1, rectangle and parallelogram reduce to the same segment.

ERectangleGauge.RemoveAllSkipRanges

Removes all the skip ranges previously created by a call to [ERectangleGauge::AddSkipRange](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void RemoveAllSkipRanges (  
    )
```

ERectangleGauge.RemoveSkipRange

After a call to [ERectangleGauge::AddSkipRange](#), removes the skip range with the given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void RemoveSkipRange (
    uint index
)
```

Parameters

index

Index of the skip range to remove, as returned by [ERectangleGauge::AddSkipRange](#).

ERectangleGauge.SamplingStep

Approximate distance between sampled points during a model fitting operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float SamplingStep
{ get; set; }
```

Remarks

Irrelevant in case of a point location operation.

To fit a model, a series of point location operations are performed along the model. The point location gauges spacing is given by the sampling step.

By default, the sampling step is set to **5**, which means 5 pixels when the field of view is not calibrated, and 5 physical units in case of a calibrated field of view.

Be aware that if the sampling step is too large, the number of sampled points along the model will not be sufficient enough to accurately locate it.

ERectangleGauge.SetMinNumFitSamples

Sets the minimum number of samples required for fitting on each side of the shape.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetMinNumFitSamples (
    int side0,
    int side1,
    int side2,
    int side3
)
```

Parameters

side0

Minimum number of samples on the *top side* of the rectangle. The default value is **2**.

side1

Minimum number of samples on the *left side* of the rectangle. If this value is not specified, it is equal to **n32Side0**. The default value is **2**.

side2

Minimum number of samples on the *bottom side* of the rectangle. If this value is not specified, it is equal to **n32Side0**. The default value is **2**.

side3

Minimum number of samples on the *right side* of the rectangle. If this value is not specified, it is equal to **n32Side1**. The default value is **2**.

Remarks

When the [ERectangleGauge::Measure](#) method is called, and if not enough valid points are detected, then the method returns directly, and the measured gauge is set to the nominal parameters.

ERectangleGauge.Smoothing

Number of pixels used for the low-pass filtering operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint Smoothing
```

```
{ get; set; }
```

Remarks

To reduce the effect of noise, the profile data can be low-pass filtered along the point location gauge direction.

ERectangleGauge.Thickness

Number of parallel segments used to extract the data profile.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint Thickness
```

```
{ get; set; }
```

Remarks

To reduce the effect of noise and/or strengthen a transition, several parallel profiles can be accumulated.

ERectangleGauge.Threshold

Threshold level used to delimit significant peaks in the data profile.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint Threshold
```

```
{ get; set; }
```

Remarks

When analyzing a derivative profile, a peak is made up of consecutive pixel values above **Threshold**. To detect weak [strong] transitions, lower [raise] the **Threshold** value.

To avoid interference of noise, an additional parameter is provided. The **MinAmplitude** parameter is an offset added to **Threshold** when a peak is to be detected.

When the pixel values of the derivative profile do not reach **Threshold + MinAmplitude**, the peak is not taken into account. Anyway, when a peak is taken into account, all the pixels with values above **Threshold** are considered (for more accuracy). Setting the **MinAmplitude** value to **0** merely cancels its effect.

ERectangleGauge.Tolerance

Searching area half thickness of the rectangle fitting gauge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
float Tolerance  
  
{ get; set; }
```

Remarks

A rectangle fitting gauge is fully defined knowing its nominal position (its center coordinates), its nominal size, its rotation angle, and its outline tolerance.

By default, the searching area thickness of the rectangle fitting gauge is **20** (2x10), which means 20 pixels when the field of view is not calibrated, and 20 physical units in case of a calibrated field of view.

ERectangleGauge.TransitionChoice

Transition choice.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
```

```
Euresys.Open_eVision_2_10.ETransitionChoice TransitionChoice
```

```
{ get; set; }
```

Remarks

Several peaks may be detected along a point location gauge. This parameter helps to select the desired transition. In case of [NthFromBegin](#) or [NthFromEnd](#) transition choice, set [ERectangleGauge::TransitionIndex](#) to specify the desired transition.

By default, the selected transition corresponds to the one with the largest amplitude ([LargestAmplitude](#)).

ERectangleGauge.TransitionIndex

Index (from **0** on) of the transition to be retained when the transition choice parameter is set to [NthFromBegin](#) or [NthFromEnd](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint TransitionIndex
```

```
{ get; set; }
```

Remarks

Several peaks may be detected along a point location gauge. This parameter helps to select the desired transition. By default, the first transition is retained (the index value is **0**).

ERectangleGauge.TransitionType

Transition type.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.ETransitionType TransitionType  
  
{ get; set; }
```

Remarks

The type of a transition tells whether it crosses increasing or decreasing gray-level values. This helps discriminate between nearby edges of an object.

By default, the searched transition type is indifferently a black to white or a white to black transition ([BwOrWb](#)).

ERectangleGauge.Type

Shape type.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
override Euresys.Open_eVision_2_10.EShapeType Type  
  
{ get; }
```

ERectangleGauge.Valid

Flag indicating if at least one valid transition has been found.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
bool Valid
```

```
{ get; }
```

Remarks

A **FALSE** value means that no measurement has been performed.

A **TRUE** value means that a transition was found along the sample path inspected with the last call to [ERectangleGauge::MeasureSample](#), and thus a point has been measured.

4.162. ERectangleRegion Class

Manages a complete context for an [ERegion](#) shaped like a rectangle.

Base Class: [ERegion](#)

Namespace: Euresys.Open_eVision_2_10

Properties

Angle	Angle of the region
Center	Center of the region
Height	Height of the region
Width	M Width of the region

Methods

Drag

e

Moves the specified handle to a new position and updates all placement parameters of the region.

[ERectangleRegion](#)

Constructs an [ERectangleRegion](#) context.

[HitTest](#)

Detects if the cursor is placed over one of the dragging handles.

[Load](#)

Loads the [ERectangleRegion](#). The given [ESerializer](#) must have been created for reading.

[operator!=](#)

Checks if this [ERectangleRegion](#) instance is not strictly equal to another

[operator=](#)

Assignment operator.

[operator==](#)

Checks if this [ERectangleRegion](#) instance is strictly equal to another

[Rotate](#)

Creates a new [ERectangleRegion](#) by rotating the [ERectangleRegion](#) around its center.

[Save](#)

Saves the [ERectangleRegion](#). The given [ESerializer](#) must have been created for writing.

[Scale](#)

Creates a new [ERectangleRegion](#) by scaling the [ERectangleRegion](#). The center of the [ERectangleRegion](#) remains at the same place.

Translate

E Creates a new [ERectangleRegion](#) by translating the [ERectangleRegion](#).
R

ERectangleRegion.Angle

Angle of the region

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Angle  
{ get; set; }
```

ERectangleRegion.Center

Center of the region

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint Center  
{ get; set; }
```

ERectangleRegion.Drag

Moves the specified handle to a new position and updates all placement parameters of the region.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Drag(
    int x,
    int y,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)
```

Parameters

x

x-coordinate of the mouse cursor.

y

y-coordinate of the mouse cursor.

zoomX

Horizontal zoom factor. By default, true scale is used.

zoomY

Vertical zoom factor. By default, true scale is used.

panX

Horizontal pan offset. By default, no pan is added.

panY

Vertical pan offset. By default, no pan is added.

Remarks

If zooming and/or panning were used when drawing the region, the same values must be used with [ERectangleRegion::HitTest](#) and [ERectangleRegion::Drag](#).

ERectangleRegion.ERectangleRegion

Constructs an [ERectangleRegion](#) context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void ERectangleRegion(
)

void ERectangleRegion(
    float originX,
    float originY,
    float width,
    float height
)

void ERectangleRegion(
    Euresys.Open_eVision_2_10.EPoint origin,
    float width,
    float height
)

void ERectangleRegion(
    float centerX,
    float centerY,
    float width,
    float height,
    float angle
)

void ERectangleRegion(
    Euresys.Open_eVision_2_10.EPoint center,
    float width,
    float height,
    float angle
)

void ERectangleRegion(
    Euresys.Open_eVision_2_10.ERectangle rectangle
)
```

```
void ERectangleRegion(  
    Euresys.Open_eVision_2_10.ERectangleRegion other  
)
```

Parameters

originX

The abscissa of the [ERectangleRegion](#)'s top left corner.

originY

The ordinate of the [ERectangleRegion](#)'s top left corner.

width

The width of the [ERectangleRegion](#).

height

The height of the [ERectangleRegion](#).

origin

The top left corner of the [ERectangleRegion](#).

centerX

The abscissa of the [ERectangleRegion](#) center.

centerY

The ordinate of the [ERectangleRegion](#) center.

angle

The angle of the [ERectangleRegion](#).

center

The center of the [ERectangleRegion](#).

rectangle

The result of an [ERectangleGauge](#) object.

other

[ERectangleRegion](#) context to copy.

Remarks

A [ERectangleRegion](#) aligned with the image axes is defined from the top left corner. Otherwise, an oriented [ERectangleRegion](#) is defined from its center point.

ERectangleRegion.Height

Height of the region

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
float Height  
  
{ get; set; }
```

ERectangleRegion.HitTest

Detects if the cursor is placed over one of the dragging handles.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
Euresys.Open_eVision_2_10.EEditionMode HitTest(  
    int x,  
    int y,  
    float zoomX,  
    float zoomY,  
    int panX,  
    int panY  
)
```

Parameters

x

x-coordinate of the mouse cursor.

y

y-coordinate of the mouse cursor.

zoomX

Horizontal zoom factor. By default, true scale is used.

zoomY

Vertical zoom factor. By default, true scale is used.

panX

Horizontal pan offset. By default, no pan is added.

panY

Vertical pan offset. By default, no pan is added.

Remarks

Returns a handle identifier, as defined by [EEditionMode](#).

If zooming and/or panning were used when drawing the region, the same values must be used with [ERectangleRegion::HitTest](#) and [ERectangleRegion::Drag](#).

ERectangleRegion.Load

Loads the [ERectangleRegion](#). The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Load(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The serializer.

ERectangleRegion.operator!=

Checks if this [ERectangleRegion](#) instance is not strictly equal to another

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
bool operator!=(  
    Euresys.Open_eVision_2_10.ERectangleRegion other  
)
```

Parameters

other

Reference to the other [ERectangleRegion](#) instance

ERectangleRegion.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.ERectangleRegion operator=(  
    Euresys.Open_eVision_2_10.ERectangleRegion other  
)
```

Parameters

other

Reference to the [ERectangleRegion](#) used for the assignment

ERectangleRegion.operator==

Checks if this [ERectangleRegion](#) instance is strictly equal to another

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool operator==(
    Euresys.Open_eVision_2_10.ERectangleRegion other
)
```

Parameters

other

Reference to the other [ERectangleRegion](#) instance

ERectangleRegion.Rotate

Creates a new [ERectangleRegion](#) by rotating the [ERectangleRegion](#) around its center.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ERectangleRegion Rotate(
    float angle
)
```

Parameters

angle

rotation angle

ERectangleRegion.Save

Saves the [ERectangleRegion](#). The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The serializer.

ERectangleRegion.Scale

Creates a new [ERectangleRegion](#) by scaling the [ERectangleRegion](#). The center of the [ERectangleRegion](#) remains at the same place.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ERectangleRegion Scale(
    float scale
)

Euresys.Open_eVision_2_10.ERectangleRegion Scale(
    float scaleX,
    float scaleY
)
```

Parameters

scale

Isotropic scale

scaleX

Horizontal scale

scaleY

Vertical scale

ERectangleRegion.Translate

Creates a new [ERectangleRegion](#) by translating the [ERectangleRegion](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ERectangleRegion Translate(
    float dx,
    float dy
)
```

Parameters

dx
Horizontal translation in pixel value

dy
Vertical translation in pixel value

ERectangleRegion.Width

Width of the region

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float Width
{ get; set; }
```

4.163. ERectangleShape Class

Manages a rectangle shape.

Base Class: [EShape](#)

Derived Class(es): [EBarcode](#) [ERectangleGauge](#)

Namespace: Euresys.Open_eVision_2_10

Properties

Angle	Orientation of the shape.
Center	Center point of the shape.
CenterX	Abscissa of the origin point of the shape.
CenterY	Ordinate of the origin point of the shape.
Rectangle	Sets the parameters of the rectangle according to a known ERectangle object.
Scale	Horizontal sensor resolution, in pixels per unit.
SizeX	X size of the ERectangleShape

SizeY

Y size of the [ERectangleShape](#)

Type

M Shape type.

e

Methods

Closest

Find the daughter shape that is the closest to this shape. To retrieve the closest shape, use [EShape::ClosestShape](#).

CopyTo

Copies all the data of the current [ERectangleShape](#) object into another [ERectangleShape](#) object and returns it.

Drag

Moves a handle to a new position and updates the position parameters of the shape.

Draw

Draws a graphical representation of a shape, as defined by [EDrawingMode](#).

DrawWithCurrentPen

Draws a graphical representation of a shape, as defined by [EDrawingMode](#).

GetCorners

Retrieves the coordinates of each corner of a [ERectangleShape](#) object.

GetEdges

Retrieves each edge of a [ERectangleShape](#) object.

GetMidEdges

Retrieves the center coordinates of each edge of a [ERectangleShape](#) object.

GetPoint

Returns the coordinates of a particular point, specified by its location in the [ERectangleShape](#) area.

HitTest

Checks if there is a handle under the cursor.

operator=

Copies all the data from another [ERectangleShape](#) object into the current [ERectangleShape](#) object

SetCenterXY

Sets the center coordinates of a [ERectangleShape](#) object.

SetFromOppositeCorners

Sets the geometric parameters of an [ERectangleShape](#) object using two opposed corners.

SetFromOriginMiddleEnd

DEPRECATED (you should use [ERectangleShape::SetFromThreeCorners](#)): Sets the geometric parameters of an [ERectangleShape](#) object using three corners.

SetFromThreeCorners

Sets the geometric parameters of an [ERectangleShape](#) object using three corners.

SetFromTwoPoints

DEPRECATED (you should use [ERectangleShape::SetFromOppositeCorners](#)): Sets the geometric parameters of an [ERectangleShape](#) object using two opposed corners.

SetSize

E Sets the size of a [ERectangleShape](#) object.

R

ERectangleShape.Angle

Orientation of the shape.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Angle  
{ get; set; }
```

ERectangleShape.Center

Center point of the shape.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint Center  
{ get; set; }
```

ERectangleShape.CenterX

Abscissa of the origin point of the shape.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float CenterX  
    { get; }
```

ERectangleShape.CenterY

Ordinate of the origin point of the shape.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float CenterY  
    { get; }
```

ERectangleShape.Closest

Find the daughter shape that is the closest to this shape. To retrieve the closest shape, use [EShape::ClosestShape](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Closest(  
    )
```

ERectangleShape.CopyTo

Copies all the data of the current [ERectangleShape](#) object into another [ERectangleShape](#) object and returns it.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ERectangleShape CopyTo(
    Euresys.Open_eVision_2_10.ERectangleShape dest,
    bool bRecursive
)
```

Parameters

dest

Pointer to the [ERectangleShape](#) object in which the current [ERectangleShape](#) object data have to be copied.

bRecursive

TRUE if the children shapes have to be copied as well, **FALSE** otherwise.

Remarks

In case of a **NULL** pointer, a new [ERectangleShape](#) object will be created and returned.

ERectangleShape.Drag

Moves a handle to a new position and updates the position parameters of the shape.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Drag(
    int n32CursorX,
    int n32CursorY
)
```

Parameters

n32CursorX

Current cursor coordinates.

n32CursorY

Current cursor coordinates.

ERectangleShape.Draw

Draws a graphical representation of a shape, as defined by [EDrawingMode](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingMode

Indicates how the point location or model fitting gauge must be displayed, as defined by [EDrawingMode](#).

daughters

TRUE if the daughters gauges are to be displayed also.

color

-

ERectangleShape.DrawWithCurrentPen

Draws a graphical representation of a shape, as defined by [EDrawingMode](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawWithCurrentPen(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingMode

Indicates how the point location or model fitting gauge must be displayed, as defined by [EDrawingMode](#).

daughters

TRUE if the daughters gauges are to be displayed also.

ERectangleShape.GetCorners

Retrieves the coordinates of each corner of a [ERectangleShape](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetCorners (
    Euresys.Open_eVision_2_10.EPoint xy,
    Euresys.Open_eVision_2_10.EPoint XXy,
    Euresys.Open_eVision_2_10.EPoint xYY,
    Euresys.Open_eVision_2_10.EPoint XYY
)
```

Parameters

xy

Coordinates of the lower leftmost corner of the [ERectangleShape](#) object.

XXy

Coordinates of the lower rightmost corner of the [ERectangleShape](#) object.

xYY

Coordinates of the upper leftmost corner of the [ERectangleShape](#) object.

XYY

Coordinates of the upper rightmost corner of the [ERectangleShape](#) object.

ERectangleShape.GetEdges

Retrieves each edge of a [ERectangleShape](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetEdges (
    Euresys.Open_eVision_2_10.ELine x,
    Euresys.Open_eVision_2_10.ELine XX,
    Euresys.Open_eVision_2_10.ELine y,
    Euresys.Open_eVision_2_10.ELine YY
)
```

Parameters

x

Leftmost edge of the [ERectangleShape](#) object.

XX

Rightmost edge of the [ERectangleShape](#) object.

Y

Lower edge of the [ERectangleShape](#) object.

YY

Upper edge of the [ERectangleShape](#) object.

ERectangleShape.GetMidEdges

Retrieves the center coordinates of each edge of a [ERectangleShape](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetMidEdges (
    Euresys.Open_eVision_2_10.EPoint x,
    Euresys.Open_eVision_2_10.EPoint XX,
    Euresys.Open_eVision_2_10.EPoint y,
    Euresys.Open_eVision_2_10.EPoint YY
)
```

Parameters

x

Center coordinates of the leftmost edge of the [ERectangleShape](#) object.

XX

Center coordinates of the rightmost edge of the [ERectangleShape](#) object.

Y

Center coordinates of the lower edge of the [ERectangleShape](#) object.

YY

Center coordinates of the upper edge of the [ERectangleShape](#) object.

ERectangleShape.GetPoint

Returns the coordinates of a particular point, specified by its location in the [ERectangleShape](#) area.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint GetPoint(
    float fractionX,
    float fractionY
)
```

Parameters

fractionX

Point location expressed as a fraction of the [ERectangleShape](#) vertical edges (range -1, +1).

fractionY

Point location expressed as a fraction of the [ERectangleShape](#) horizontal edges (range -1, +1).

ERectangleShape.HitTest

Checks if there is a handle under the cursor.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool HitTest(
    bool bDaughters
)
```

Parameters

bDaughters

Indicates if the check must be done in the whole hierarchy or just this object.

ERectangleShape.operator=

Copies all the data from another ERectangleShape object into the current ERectangleShape object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ERectangleShape operator=(
    Euresys.Open_eVision_2_10.ERectangleShape other
)
```

Parameters

other

ERectangleShape object to be copied

ERectangleShape.Rectangle

Sets the parameters of the rectangle according to a known [ERectangle](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
virtual Euresys.Open_eVision_2_10.ERectangle Rectangle
{ get; set; }
```

ERectangleShape.Scale

Horizontal sensor resolution, in pixels per unit.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Scale  
  
{ get; set; }
```

ERectangleShape.SetCenterXY

Sets the center coordinates of a [ERectangleShape](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void SetCenterXY(  
    float centerX,  
    float centerY  
)
```

Parameters

centerX

Center coordinates of the [ERectangleShape](#) object.

centerY

Center coordinates of the [ERectangleShape](#) object.

ERectangleShape.SetFromOppositeCorners

Sets the geometric parameters of an [ERectangleShape](#) object using two opposed corners.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetFromOppositeCorners(
    Euresys.Open_eVision_2_10.EPoint origin,
    Euresys.Open_eVision_2_10.EPoint end
)
```

Parameters

origin

Origin point coordinates of the rectangle.

end

End point coordinates of the rectangle.

Remarks

In a direct coordinate system, the abscissa extends rightwards and the ordinate extends upwards. The coordinate system is said to be inverse if the abscissa extends rightwards and the ordinate extends downwards.

ERectangleShape.SetFromOriginMiddleEnd

DEPRECATED (you should use [ERectangleShape::SetFromThreeCorners](#)): Sets the geometric parameters of an [ERectangleShape](#) object using three corners.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void SetFromOriginMiddleEnd(  
    Euresys.Open_eVision_2_10.EPoint origin,  
    Euresys.Open_eVision_2_10.EPoint middle,  
    Euresys.Open_eVision_2_10.EPoint end  
)
```

Parameters

origin

Origin point coordinates of the rectangle.

middle

Middle point coordinates of the rectangle.

end

End point coordinates of the rectangle.

Remarks

In a direct coordinate system, the abscissa extends rightwards and the ordinate extends upwards. The coordinate system is said to be inverse if the abscissa extends rightwards and the ordinate extends downwards.

ERectangleShape.SetFromThreeCorners

Sets the geometric parameters of an [ERectangleShape](#) object using three corners.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void SetFromThreeCorners(  
    Euresys.Open_eVision_2_10.EPoint origin,  
    Euresys.Open_eVision_2_10.EPoint middle,  
    Euresys.Open_eVision_2_10.EPoint end  
)
```

Parameters

origin

Origin point coordinates of the rectangle.

middle

Middle point coordinates of the rectangle.

end

End point coordinates of the rectangle.

Remarks

In a direct coordinate system, the abscissa extends rightwards and the ordinate extends upwards. The coordinate system is said to be inverse if the abscissa extends rightwards and the ordinate extends downwards.

ERectangleShape.SetFromTwoPoints

DEPRECATED (you should use [ERectangleShape::SetFromOppositeCorners](#)): Sets the geometric parameters of an [ERectangleShape](#) object using two opposed corners.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetFromTwoPoints(
    Euresys.Open_eVision_2_10.EPoint origin,
    Euresys.Open_eVision_2_10.EPoint end
)
```

Parameters

origin

Origin point coordinates of the rectangle.

end

End point coordinates of the rectangle.

Remarks

In a direct coordinate system, the abscissa extends rightwards and the ordinate extends upwards. The coordinate system is said to be inverse if the abscissa extends rightwards and the ordinate extends downwards.

ERectangleShape.SetSize

Sets the size of a [ERectangleShape](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void SetSize(  
    float sizeX,  
    float sizeY  
)
```

Parameters

sizeX

Nominal size X of the [ERectangleShape](#) object. Default values is **100**.

sizeY

Nominal size Y of the [ERectangleShape](#) object. Default values is **100**.

Remarks

A [ERectangleShape](#) object is fully defined knowing its nominal position (given by the coordinates of its center), its nominal size, its rotation angle and its outline tolerance. By default, the width and height values are **100**, which means 100 pixels when the field of view is not calibrated and 100 "units" in case of a calibrated field of view.

ERectangleShape.SizeX

X size of the [ERectangleShape](#)

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
float SizeX  
{ get; }
```

ERectangleShape.SizeY

Y size of the [ERectangleShape](#)

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float SizeY  
  
{ get; }
```

ERectangleShape.Type

Shape type.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
override Euresys.Open_eVision_2_10.EShapeType Type  
  
{ get; }
```

4.164. ERectangularCropper Class

Manages a point cloud cropper in the shape of a rectangular parallelepiped.

Namespace: Euresys.Open_eVision_2_10.Easy3D

Methods

Crop

Crops a [EPointCloud](#).

[ERectangularCropper](#)

Creates an [ERectangularCropper](#) object.

operator=

| E Assignment operator.

R

ERectangularCropper.Crop

Crops a [EPointCloud](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Crop(
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud cloudIn,
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud cloudOut,
    bool invertCrop
)
```

Parameters

cloudIn

Cloud to be cropped.

cloudOut

Cropped cloud.

invertCrop

Indicates if the points kept must be the points inside (true) or outside (false) the rectangular parallelepiped.

ERectangularCropper.ERectangularCropper

Creates an [ERectangularCropper](#) object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ERectangularCropper (
    ref Euresys.Open_eVision_2_10.Easy3D.E3DPoint center,
    float xSize,
    float ySize,
    float zSize,
    float roll,
    float pitch,
    float yaw
)

void ERectangularCropper (
    Euresys.Open_eVision_2_10.Easy3D.ERectangularCropper other
)
```

Parameters

center

Center of the rectangular parallelepiped.

xSize

Size along the X axis before rotation of the rectangular parallelepiped.

ySize

Size along the Y axis before rotation of the rectangular parallelepiped.

zSize

Size along the Z axis before rotation of the rectangular parallelepiped.

roll

Roll (rotation along the X axis) of the rectangular parallelepiped.

pitch

Pitch (rotation along the Y axis) of the rectangular parallelepiped.

yaw

Yaw (rotation along the Z axis) of the rectangular parallelepiped.

other

Reference [ERectangularCropper](#) used for the initialization.

ERectangularCropper.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.ERectangularCropper operator=(
    Euresys.Open_eVision_2_10.Easy3D.ERectangularCropper other
)
```

Parameters

other

An other [ERectangularCropper](#).

4.165. EReferenceImageSegmenter Class

Segments an image using a pixel-by-pixel single threshold given as an image.

Remarks

This segmenter is applicable to [EROIBW8](#), [EROIBW16](#) and [EROIC24](#) images. It produces coded images with two layers. The threshold is defined for each pixel individually by means of a reference image of the same type as the source image.

For grayscale images, the White layer (usually, with index 1) contains unmasked pixels having a gray value in a range defined by the gray value of the respective pixel in the reference image and the white color.

For RGB color images, the White layer (usually, with index 1) contains unmasked pixels having a color inside the cube of the RGB color space defined by the color of the respective pixel in the reference image and the white color (255,255,255).

The Black layer (usually, with index 0) contains the remaining unmasked pixels.

Base Class: [ETwoLayersImageSegmenter](#)

Namespace: Euresys.Open_eVision_2_10.Segmenters

Properties

[BlackLayerEncoded](#)

Black layer encoding status.

BlackLayerIndex

Index of the black layer in the destination coded image.

ReferenceImageBW16

Reference image for the segmentation of [EROIBW16](#) images.

ReferenceImageBW8

Reference image for the segmentation of [EROIBW8](#) images.

ReferenceImageC24

Reference image for the segmentation of [EROIC24](#) images.

WhiteLayerEncoded

White layer encoding status.

WhiteLayerIndex

Index of the white layer in the destination coded image.

R

ReferenceImageSegmenter.BlackLayerEncoded

Black layer encoding status.

Namespace: Euresys.Open_eVision_2_10.Segmenters

[C#]

```
override bool BlackLayerEncoded
{ get; set; }
```

EReferenceImageSegmenter.BlackLayerIndex

Index of the black layer in the destination coded image.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]  
override uint BlackLayerIndex  
  
    { get; set; }
```

Remarks

Setting this property automatically switches on the encoding of the black layer.

EReferenceImageSegmenter.ReferenceImageBW16

Reference image for the segmentation of [EROIBW16](#) images.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]  
Euresys.Open_eVision_2_10.EROIBW16 ReferenceImageBW16  
  
    { get; set; }
```

EReferenceImageSegmenter.ReferenceImageBW8

Reference image for the segmentation of [EROIBW8](#) images.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]
```

```
Euresys.Open_eVision_2_10.EROIBW8 ReferenceImageBW8
```

```
{ get; set; }
```

EReferenceImageSegmenter.ReferenceImageC24

Reference image for the segmentation of [EROIC24](#) images.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]
```

```
Euresys.Open_eVision_2_10.EROIC24 ReferenceImageC24
```

```
{ get; set; }
```

EReferenceImageSegmenter.WhiteLayerEncoded

White layer encoding status.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]
```

```
override bool WhiteLayerEncoded
```

```
{ get; set; }
```

EReferenceImageSegmenter.WhiteLayerIndex

Index of the white layer in the destination coded image.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]  
override uint WhiteLayerIndex  
  
    { get; set; }
```

Remarks

Setting this property automatically switches on the encoding of the white layer.

4.166. ERegion Class

Manages a complete context for a [ERegion](#) (Arbitrary Shaped ROI)

Derived Class(es): [EEllipseRegion](#) [ECircleRegion](#) [EPolygonRegion](#) [ERectangleRegion](#)

Namespace: Euresys.Open_eVision_2_10

Properties

[BoundingBoxHeight](#)

Bounding box height.

[BoundingBoxOrgX](#)

Bounding box origin abscissa.

[BoundingBoxOrgY](#)

Bounding box origin ordinate.

BoundingBoxWidth

Bounding box width.

EditionMode

Graphical edition mode

Runs

M Runs of the region

e

thods

CropRuns

Creates a new **ERegion** by cropping the **ERegion** runs within a given area.

Drag

Moves the specified handle to a new position and updates all placement parameters of the region.

Draw

Draws the **ERegion** shape.

DrawContour

Draws the **ERegion** contour.

DrawContourWithCurrentPen

Draws the **ERegion** contour with the current pen.

DrawHandles

Draws the region handles.

DrawHandlesWithCurrentPen

Draws the region handles with the current pen.

DrawWithCurrentPen

Draws the [ERegion](#) area.

[ERegion](#)

Constructs an [ERegion](#) context.

HitTest

Detects if the cursor is placed over one of the dragging handles.

Intersection

Creates a new [ERegion](#) by intersecting two [ERegion](#).

Load

Loads the [ERegion](#). The given [ESerializer](#) must have been created for reading.

operator!=

Checks if this [ERegion](#) instance is not strictly equal to another (order of the runs included)

operator=

Assignment operator.

operator==

Checks if this [ERegion](#) instance is strictly equal to another (order of the runs included)

Prepare

Computes the values necessary for the [ERegion](#) to be used during processing.

Save

Saves the [ERegion](#). The given [ESerializer](#) must have been created for writing.

Subtraction

Creates a new [ERegion](#) by subtracting one [ERegion](#) from another.

ToImage

Exports the [ERegion](#) to a mask-type image.

Union

Creates a new [ERegion](#) by combining two [ERegion](#).

R

egion.BoundingBoxHeight

Bounding box height.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
virtual int BoundingBoxHeight  
    { get; }
```

Remarks

The height is the height of the upright rectangle encompassing the [ERun](#).

ERegion.BoundingBoxOrgX

Bounding box origin abscissa.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
virtual int BoundingBoxOrgX
```

```
{ get; }
```

Remarks

The origin is the top left corner of the upright rectangle encompassing the [ERun](#).

ERegion.BoundingBoxOrgY

Bounding box origin ordinate.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
virtual int BoundingBoxOrgY  
  
{ get; }
```

Remarks

The origin is the top left corner of the upright rectangle encompassing the [ERun](#).

ERegion.BoundingBoxWidth

Bounding box width.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
virtual int BoundingBoxWidth  
  
{ get; }
```

Remarks

The width is the width of the upright rectangle encompassing the [ERun](#).

ERegion.CropRuns

Creates a new [ERegion](#) by cropping the [ERegion](#) runs within a given area.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ERegion CropRuns (
    int orgX,
    int orgY,
    int width,
    int height
)
```

Parameters

orgX

X origin of the cropping area.

orgY

Y origin of the cropping area.

width

Width of the cropping area.

height

height of the cropping area.

ERegion.Drag

Moves the specified handle to a new position and updates all placement parameters of the region.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void Drag(  
    int x,  
    int y,  
    float zoomX,  
    float zoomY,  
    int panX,  
    int panY  
)
```

Parameters

x

x-coordinate of the mouse cursor.

y

y-coordinate of the mouse cursor.

zoomX

Horizontal zoom factor. By default, true scale is used.

zoomY

Vertical zoom factor. By default, true scale is used.

panX

Horizontal pan offset. By default, no pan is added.

panY

Vertical pan offset. By default, no pan is added.

Remarks

If zooming and/or panning were used when drawing the region, the same values must be used with [ERegion::HitTest](#) and [ERegion::Drag](#).

ERegion.Draw

Draws the [ERegion](#) shape.

Namespace: Euresys.Open_eVision_2_10

```

[C#]
void Draw(
    IntPtr graphicContext,
    float opacity,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    float opacity,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)

```

Parameters

graphicContext

Handle of the device context on which to draw.

opacity

Opacity of the drawn area (range: 0.0 to 1.0).

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

color

The color in which to draw the [ERegion](#).

ERegion.DrawContour

Draws the [ERegion](#) contour.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawContour(
    IntPtr graphicContext,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)

void DrawContour(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

color

The color in which to draw the [ERegion](#).

ERegion.DrawContourWithCurrentPen

Draws the [ERegion](#) contour with the current pen.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawContourWithCurrentPen (
    IntPtr hdc,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)
```

Parameters

hdc

Handle of the device context on which to draw.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

ERegion.DrawHandles

Draws the region handles.

Namespace: Euresys.Open_eVision_2_10

```
[C#]

void DrawHandles (
    IntPtr hdc,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)

void DrawHandles (
    IntPtr hdc,
    Euresys.Open_eVision_2_10.ERGBColor color,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)

void DrawHandles (
    Euresys.Open_eVision_2_10.EDrawAdapter drawAdapter,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)
```

Parameters

hdc

Handle of the device context on which to draw.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

color

The color in which to draw the region.

drawAdapter

-

ERegion.DrawHandlesWithCurrentPen

Draws the region handles with the current pen.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawHandlesWithCurrentPen (
    IntPtr hdc,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)
```

Parameters

hdc

Handle of the device context on which to draw.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

ERegion.DrawWithCurrentPen

Draws the [ERegion](#) area.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawWithCurrentPen(
    IntPtr graphicContext,
    float opacity,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

opacity

Opacity of the drawn area (range: 0.0 to 1.0).

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

ERegion.EditionMode

Graphical edition mode

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EEditionMode EditionMode  
  
{ get; set; }
```

ERegion.ERegion

Constructs an [ERegion](#) context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void ERegion(  
)  
  
void ERegion(  
    Euresys.Open_eVision_2_10.ERegion other  
)  
  
void ERegion(  
    Euresys.Open_eVision_2_10.EROIBW8 roi,  
    Euresys.Open_eVision_2_10.EBW8 threshold  
)  
  
void ERegion(  
    Euresys.Open_eVision_2_10.EROIBW16 roi,  
    Euresys.Open_eVision_2_10.EBW16 threshold  
)  
  
void ERegion(  
    Euresys.Open_eVision_2_10.EMatchPosition position,  
    int modelWidth,  
    int modelHeight  
)  
  
void ERegion(  
    Euresys.Open_eVision_2_10.EFoundPattern pattern  
)
```

```
void ERegion(  
    Euresys.Open_eVision_2_10.ECodedElement codedElement  
)  
  
void ERegion(  
    Euresys.Open_eVision_2_10.ERun[] runs  
)
```

Parameters

other

[ERegion](#) context to copy.

roi

Mask ROI.

threshold

Mask Threshold.

position

Result of an EasyMatch process.

modelWidth

Width of an EasyMatch model.

modelHeight

Height of an EasyMatch model.

pattern

Result of an EasyFind process.

codedElement

Result of an EasyObject process.

runs

List of [ERun](#).

ERegion.HitTest

Detects if the cursor is placed over one of the dragging handles.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EEditionMode HitTest(
    int x,
    int y,
    float zoomX,
    float zoomY,
    int panX,
    int panY
)
```

Parameters

x

x-coordinate of the mouse cursor.

y

y-coordinate of the mouse cursor.

zoomX

Horizontal zoom factor. By default, true scale is used.

zoomY

Vertical zoom factor. By default, true scale is used.

panX

Horizontal pan offset. By default, no pan is added.

panY

Vertical pan offset. By default, no pan is added.

Remarks

Returns a handle identifier, as defined by [EEditionMode](#).

If zooming and/or panning were used when drawing the region, the same values must be used with [ERegion::HitTest](#) and [ERegion::Drag](#).

ERegion.Intersection

Creates a new [ERegion](#) by intersecting two [ERegion](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ERegion Intersection(
    Euresys.Open_eVision_2_10.ERegion region1,
    Euresys.Open_eVision_2_10.ERegion region2
)
```

Parameters

region1

First region.

region2

Second region.

ERegion.Load

Loads the [ERegion](#). The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Load(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The serializer.

ERegion.operator!=

Checks if this [ERegion](#) instance is not strictly equal to another (order of the runs included)

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool operator!=(  
    Euresys.Open_eVision_2_10.ERegion other  
)
```

Parameters

other

Reference to the other [ERegion](#) instance

ERegion.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.ERegion operator=(  
    Euresys.Open_eVision_2_10.ERegion other  
)
```

Parameters

other

Reference to the [ERegion](#) used for the assignment.

ERegion.operator==

Checks if this [ERegion](#) instance is strictly equal to another (order of the runs included)

Namespace: Euresys.Open_eVision_2_10


```
[C#]
bool operator==(
    Euresys.Open_eVision_2_10.ERegion other
)
```

Parameters

other

Reference to the other [ERegion](#) instance

ERegion.Prepare

Computes the values necessary for the [ERegion](#) to be used during processing.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Prepare(
    Euresys.Open_eVision_2_10.EROIBW8 roi
)
void Prepare(
    Euresys.Open_eVision_2_10.EROIBW16 roi
)
void Prepare(
    Euresys.Open_eVision_2_10.EROIC24 roi
)
void Prepare(
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 zmap
)
void Prepare(
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 zmap
)
```

```

void Prepare (
    Euresys.Open_eVision_2_10.Easy3D.EZMap32f zmap
)

void Prepare (
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 dm
)

void Prepare (
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 dm
)

void Prepare (
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap32f dm
)

void Prepare (
    int width,
    int height
)

void Prepare (
    int orgX,
    int orgY,
    int width,
    int height
)

```

Parameters

roi

Destination or source [EBaseROI](#).

zmap

Destination or source [EZMap](#).

dm

Destination or source [EDepthMap](#).

width

Width of the source or destination context.

height

Height of the source or destination context.

orgX

X-Axis origin of the source or destination context.

orgY

Y-Axis origin of the source or destination context.

Remarks

This method should be called once after the [ERegion](#) has been parameterized and before the [ERegion](#) is used. If necessary, it will be done automatically before any usage but it will increase the processing time.

ERegion.Runs

Runs of the region

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.ERun[] Runs  
    { get; set; }
```

ERegion.Save

Saves the [ERegion](#). The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Save(  
    Euresys.Open_eVision_2_10.ESerializer serializer  
)
```

Parameters

serializer

The serializer.

ERegion.Subtraction

Creates a new [ERegion](#) by subtracting one [ERegion](#) from another.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ERegion Subtraction(
    Euresys.Open_eVision_2_10.ERegion region1,
    Euresys.Open_eVision_2_10.ERegion region2
)
```

Parameters

region1

Original region.

region2

Region to subtract.

ERegion.ToImage

Exports the [ERegion](#) to a mask-type image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ToImage(
    Euresys.Open_eVision_2_10.EImageC24 img,
    Euresys.Open_eVision_2_10.EC24 background,
    Euresys.Open_eVision_2_10.EC24 foreground
)
```

```
void ToImage(  
    Euresys.Open_eVision_2_10.EImageBW8 img,  
    Euresys.Open_eVision_2_10.EBW8 background,  
    Euresys.Open_eVision_2_10.EBW8 foreground  
)  
  
void ToImage(  
    Euresys.Open_eVision_2_10.EImageBW16 img,  
    Euresys.Open_eVision_2_10.EBW16 background,  
    Euresys.Open_eVision_2_10.EBW16 foreground  
)
```

Parameters

img

Destination image.

background

Background color.

foreground

Foreground color.

ERegion.Union

Creates a new [ERegion](#) by combining two [ERegion](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
Euresys.Open_eVision_2_10.ERegion Union(  
    Euresys.Open_eVision_2_10.ERegion region1,  
    Euresys.Open_eVision_2_10.ERegion region2  
)
```

Parameters

region1

First region.

region2

Second region.

4.167. EROIBW1 Class

The EROIBW1 class is used to represent rectangular regions of interest inside BW1 black and white images. See ROIs.

Base Class: [EBaseROI](#)

Derived Class(es): [EImageBW1](#)

Namespace: Euresys.Open_eVision_2_10

Properties

[FirstSubROI](#)

See GetFirstSubROI in the derived classes

[NextSiblingROI](#)

This method returns the ROI that is the next sibling of this ROI.

[Parent](#)

-

[TopParent](#)

M-

e

Methods

[EROIBW1](#)

-

[GetBitIndex](#)

-

GetNextROI

See GetNextROI in the derived classes

GetPixel

Allows reading a single pixel value in the ROI or image.

operator=

-

Serialize

Serializes the [EROIBW1](#).

SetPixel

Allows writing a single pixel value in the ROI or image.

R

OIBW1.EROIBW1

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EROIBW1 (
)
void EROIBW1 (
    Euresys.Open_eVision_2_10.EROIBW1 other
)
```

Parameters

other

-

EROIBW1.FirstSubROI

See GetFirstSubROI in the derived classes

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
new Euresys.Open_eVision_2_10.EROIBW1 FirstSubROI  
    { get; }
```

EROIBW1.GetBitIndex

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
System.UInt64 GetBitIndex(  
    int x,  
    int y  
)
```

Parameters

x

-

y

-

EROIBW1.GetNextROI

See GetNextROI in the derived classes

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EROIBW1 GetNextROI (
    Euresys.Open_eVision_2_10.EBaseROI startROI
)
```

Parameters

startROI

-

EROIBW1.GetPixel

Allows reading a single pixel value in the ROI or image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EBW1 GetPixel (
    int x,
    int y
)
```

Parameters

x

-

y

Remarks

Although coordinates outside of the ROI can be supplied, this function will raise an error condition if the coordinates are outside of the top parent image. It should be noted that calling this function several thousand times can be slow. The recommended way to access pixel content in an image or ROI is to use the [EROIBW1](#) and [EROIBW1](#) functions. For BW8 images/ROIs, using the C++ API only, it is possible to access pixels data faster through an intermediate object: [EBW8PixelAccessor](#).

EROIBW1.NextSiblingROI

This method returns the ROI that is the next sibling of this ROI.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
new Euresys.Open_eVision_2_10.EROIBW1 NextSiblingROI  
{ get; }
```

EROIBW1.operator=

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EROIBW1 operator=(  
    Euresys.Open_eVision_2_10.EROIBW1 other  
)
```

Parameters

other

-

EROIBW1.Parent

Namespace: Euresys.Open_eVision_2_10

```
[C#]
new Euresys.Open_eVision_2_10.EROIBW1 Parent
    { get; }
```

EROIBW1.Serialize

Serializes the [EROIBW1](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Serialize(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

Serializer used for the serialization.

EROIBW1.SetPixel

Allows writing a single pixel value in the ROI or image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetPixel(
    Euresys.Open_eVision_2_10.EBW1 value,
    int x,
    int y
)
```

Parameters

value

-

x

-

y

-

Remarks

Although coordinates outside of the ROI can be supplied, this function will raise an error condition if the coordinates are outside of the top parent image. It should be noted that calling this function several thousand times can be slow. The recommended way to access pixel content in an image or ROI is to use the [EROIBW1](#) and [EROIBW1](#) functions. For BW8 images/ROIs, using the C++ API only, it is possible to access pixels data faster through an intermediate object: [EBW8PixelAccessor](#).

EROIBW1.TopParent

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EImageBW1 TopParent
```

```
{ get; }
```

4.168. EROIBW16 Class

The EROIBW16 class is used to represent rectangular regions of interest inside BW16 gray-level images. See ROIs.

Base Class: [EBaseROI](#)

Derived Class(es): [EImageBW16](#)

Namespace: Euresys.Open_eVision_2_10

Properties

[FirstSubROI](#)

See GetFirstSubROI in the derived classes

[NextSiblingROI](#)

This method returns the ROI that is the next sibling of this ROI.

[Parent](#)

-

[TopParent](#)

M-

e

Methods

[EROIBW16](#)

-

GetNextROI

See GetNextROI in the derived classes

GetPixel

Allows reading a single pixel value in the ROI or image.

operator=

-

Serialize

Serializes the [EROIBW16](#).

SetPixel

Allows writing a single pixel value in the ROI or image.

R

OIBW16.EROIBW16

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EROIBW16 (
)
void EROIBW16 (
    Euresys.Open_eVision_2_10.EROIBW16 other
)
```

Parameters

other

-

EROIBW16.FirstSubROI

See GetFirstSubROI in the derived classes

Namespace: Euresys.Open_eVision_2_10

```
[C#]
new Euresys.Open_eVision_2_10.EROIBW16 FirstSubROI
    { get; }
```

EROIBW16.GetNextROI

See GetNextROI in the derived classes

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EROIBW16 GetNextROI (
    Euresys.Open_eVision_2_10.EBaseROI startROI
)
```

Parameters

startROI

-

EROIBW16.GetPixel

Allows reading a single pixel value in the ROI or image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EBW16 GetPixel(  
    int x,  
    int y  
)
```

Parameters

x

-

y

-

Remarks

Although coordinates outside of the ROI can be supplied, this function will raise an error condition if the coordinates are outside of the top parent image. It should be noted that calling this function several thousand times can be slow. The recommended way to access pixel content in an image or ROI is to use the [EROIBW16](#) and [EROIBW16](#) functions. For BW8 images/ROIs, using the C++ API only, it is possible to access pixels data faster through an intermediate object: EBW8PixelAccessor.

EROIBW16.NextSiblingROI

This method returns the ROI that is the next sibling of this ROI.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
new Euresys.Open_eVision_2_10.EROIBW16 NextSiblingROI  
    { get; }
```


EROIBW16.operator=

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EROIBW16 operator=(  
    Euresys.Open_eVision_2_10.EROIBW16 other  
)
```

Parameters

other

-

EROIBW16.Parent

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
new Euresys.Open_eVision_2_10.EROIBW16 Parent  
    { get; }
```

EROIBW16.Serialize

Serializes the [EROIBW16](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Serialize(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

Serializer used for the serialization.

EROIBW16.SetPixel

Allows writing a single pixel value in the ROI or image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetPixel(
    Euresys.Open_eVision_2_10.EBW16 value,
    int x,
    int y
)
```

Parameters

value

-

x

-

y

-

Remarks

Although coordinates outside of the ROI can be supplied, this function will raise an error condition if the coordinates are outside of the top parent image. It should be noted that calling this function several thousand times can be slow. The recommended way to access pixel content in an image or ROI is to use the [EROIBW16](#) and [EROIBW16](#) functions. For BW8 images/ROIs, using the C++ API only, it is possible to access pixels data faster through an intermediate object: `EBW8PixelAccessor`.

EROIBW16.TopParent

-

Namespace: `Euresys.Open_eVision_2_10`

```
[C#]
Euresys.Open_eVision_2_10.EImageBW16 TopParent
{ get; }
```

4.169. EROIBW32 Class

The `EROIBW32` class is used to represent rectangular regions of interest inside `BW32` gray-level images. See `ROIs`.

Base Class: [EBaseROI](#)

Derived Class(es): [EImageBW32](#)

Namespace: `Euresys.Open_eVision_2_10`

Properties

[FirstSubROI](#)

See `GetFirstSubROI` in the derived classes

NextSiblingROI

This method returns the ROI that is the next sibling of this ROI.

Parent

-

TopParent

M-

e

thods

EROIBW32

-

GetNextROI

See GetNextROI in the derived classes

GetPixel

Allows reading a single pixel value in the ROI or image.

operator=

-

Serialize

Serializes the [EROIBW32](#).

SetPixel

E Allows writing a single pixel value in the ROI or image.

R

OIBW32.EROIBW32

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EROIBW32 (
)
void EROIBW32 (
    Euresys.Open_eVision_2_10.EROIBW32 other
)
```

Parameters

other

-

EROIBW32.FirstSubROI

See GetFirstSubROI in the derived classes

Namespace: Euresys.Open_eVision_2_10

```
[C#]
new Euresys.Open_eVision_2_10.EROIBW32 FirstSubROI
{ get; }
```

EROIBW32.GetNextROI

See GetNextROI in the derived classes

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EROIBW32 GetNextROI (
    Euresys.Open_eVision_2_10.EBaseROI startROI
)
```

Parameters

startROI

-

EROIBW32.GetPixel

Allows reading a single pixel value in the ROI or image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EBW32 GetPixel (
    int x,
    int y
)
```

Parameters

x

-

y

-

Remarks

Although coordinates outside of the ROI can be supplied, this function will raise an error condition if the coordinates are outside of the top parent image. It should be noted that calling this function several thousand times can be slow. The recommended way to access pixel content in an image or ROI is to use the [EROIBW32](#) and [EROIBW32](#) functions. For BW8 images/ROIs, using the C++ API only, it is possible to access pixels data faster through an intermediate object: [EBW8PixelAccessor](#).

EROIBW32.NextSiblingROI

This method returns the ROI that is the next sibling of this ROI.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
new Euresys.Open_eVision_2_10.EROIBW32 NextSiblingROI
{ get; }
```

EROIBW32.operator=

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EROIBW32 operator=(
    Euresys.Open_eVision_2_10.EROIBW32 other
)
```

Parameters

other

-

EROIBW32.Parent

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
new Euresys.Open_eVision_2_10.EROIBW32 Parent
    { get; }
```

EROIBW32.Serialize

Serializes the [EROIBW32](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Serialize(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

Serializer used for the serialization.

EROIBW32.SetPixel

Allows writing a single pixel value in the ROI or image.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
void SetPixel(
    Euresys.Open_eVision_2_10.EBW32 value,
    int x,
    int y
)
```

Parameters

value

-

x

-

y

-

Remarks

Although coordinates outside of the ROI can be supplied, this function will raise an error condition if the coordinates are outside of the top parent image. It should be noted that calling this function several thousand times can be slow. The recommended way to access pixel content in an image or ROI is to use the [EROIBW32](#) and [EROIBW32](#) functions. For BW8 images/ROIs, using the C++ API only, it is possible to access pixels data faster through an intermediate object: `EBW8PixelAccessor`.

EROIBW32.TopParent

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EImageBW32 TopParent
{ get; }
```

4.170. EROI8 Class

The EROI8 class is used to represent rectangular regions of interest inside BW8 gray-level images. See ROIs.

Base Class: [EBaseROI](#)

Derived Class(es): [EImageBW8](#)

Namespace: Euresys.Open_eVision_2_10

Properties

[FirstSubROI](#)

See [GetFirstSubROI](#) in the derived classes

[NextSiblingROI](#)

This method returns the ROI that is the next sibling of this ROI.

[Parent](#)

-

[TopParent](#)

M-

e

Methods

[EROI8](#)

-

[GetNextROI](#)

See [GetNextROI](#) in the derived classes

[GetPixel](#)

Allows reading a single pixel value in the ROI or image.

operator=

-

Serialize

Serializes the [EROIBW8](#).

SetPixel

Allows writing a single pixel value in the ROI or image.

R

EROIBW8.EROIBW8

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EROIBW8 (
)
void EROIBW8 (
    Euresys.Open_eVision_2_10.EROIBW8 other
)
```

Parameters

other

-

EROIBW8.FirstSubROI

See GetFirstSubROI in the derived classes

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
new Euresys.Open_eVision_2_10.EROIBW8 FirstSubROI  
    { get; }
```

EROIBW8.GetNextROI

See GetNextROI in the derived classes

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EROIBW8 GetNextROI (  
    Euresys.Open_eVision_2_10.EBaseROI startROI  
)
```

Parameters

startROI

-

EROIBW8.GetPixel

Allows reading a single pixel value in the ROI or image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EBW8 GetPixel(
    int x,
    int y
)
```

Parameters

x
-
y
-

Remarks

Although coordinates outside of the ROI can be supplied, this function will raise an error condition if the coordinates are outside of the top parent image. It should be noted that calling this function several thousand times can be slow. The recommended way to access pixel content in an image or ROI is to use the [EROIBW8](#) and [EROIBW8](#) functions. For BW8 images/ROIs, using the C++ API only, it is possible to access pixels data faster through an intermediate object: EBW8PixelAccessor.

EROIBW8.NextSiblingROI

This method returns the ROI that is the next sibling of this ROI.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
new Euresys.Open_eVision_2_10.EROIBW8 NextSiblingROI
    { get; }
```

EROIBW8.operator=

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EROIBW8 operator=(  
    Euresys.Open_eVision_2_10.EROIBW8 other  
)
```

Parameters

other

-

EROIBW8.Parent

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
new Euresys.Open_eVision_2_10.EROIBW8 Parent  
    { get; }
```

EROIBW8.Serialize

Serializes the [EROIBW8](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Serialize(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

Serializer used for the serialization.

EROIBW8.SetPixel

Allows writing a single pixel value in the ROI or image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetPixel(
    Euresys.Open_eVision_2_10.EBW8 value,
    int x,
    int y
)
```

Parameters

value

-

x

-

y

-

Remarks

Although coordinates outside of the ROI can be supplied, this function will raise an error condition if the coordinates are outside of the top parent image. It should be noted that calling this function several thousand times can be slow. The recommended way to access pixel content in an image or ROI is to use the [EROIBW8](#) and [EROIBW8](#) functions. For BW8 images/ROIs, using the C++ API only, it is possible to access pixels data faster through an intermediate object: `EBW8PixelAccessor`.

EROIBW8.TopParent

-

Namespace: `Euresys.Open_eVision_2_10`

```
[C#]
Euresys.Open_eVision_2_10.EImageBW8 TopParent
{ get; }
```

4.171. EROIC15 Class

The EROIC15 class is used to represent rectangular regions of interest inside C15 color images. See ROIs.

Base Class: [EBaseROI](#)

Derived Class(es): [EImageC15](#)

Namespace: `Euresys.Open_eVision_2_10`

Properties

[FirstSubROI](#)

See `GetFirstSubROI` in the derived classes

NextSiblingROI

This method returns the ROI that is the next sibling of this ROI.

Parent

-

TopParent

M-

e

thods

EROIC15

-

GetNextROI

See GetNextROI in the derived classes

GetPixel

Allows reading a single pixel value in the ROI or image.

operator=

-

Serialize

Serializes the [EROIC15](#).

SetPixel

E Allows writing a single pixel value in the ROI or image.

R

OIC15.EROIC15

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EROIC15 (
)
void EROIC15 (
    Euresys.Open_eVision_2_10.EROIC15 other
)
```

Parameters

other

-

EROIC15.FirstSubROI

See GetFirstSubROI in the derived classes

Namespace: Euresys.Open_eVision_2_10

```
[C#]
new Euresys.Open_eVision_2_10.EROIC15 FirstSubROI
{ get; }
```

EROIC15.GetNextROI

See GetNextROI in the derived classes

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EROIC15 GetNextROI (
    Euresys.Open_eVision_2_10.EBaseROI startROI
)
```

Parameters

startROI

-

EROIC15.GetPixel

Allows reading a single pixel value in the ROI or image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EC15 GetPixel (
    int x,
    int y
)
```

Parameters

x

-

y

-

Remarks

Although coordinates outside of the ROI can be supplied, this function will raise an error condition if the coordinates are outside of the top parent image. It should be noted that calling this function several thousand times can be slow. The recommended way to access pixel content in an image or ROI is to use the [EROIC15](#) and [EROIC15](#) functions. For BW8 images/ROIs, using the C++ API only, it is possible to access pixels data faster through an intermediate object: [EBW8PixelAccessor](#).

EROIC15.NextSiblingROI

This method returns the ROI that is the next sibling of this ROI.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
new Euresys.Open_eVision_2_10.EROIC15 NextSiblingROI  
    { get; }
```

EROIC15.operator=

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EROIC15 operator=(  
    Euresys.Open_eVision_2_10.EROIC15 other  
)
```

Parameters

other

-

EROIC15.Parent

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
new Euresys.Open_eVision_2_10.EROIC15 Parent
    { get; }
```

EROIC15.Serialize

Serializes the [EROIC15](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Serialize(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

Serializer used for the serialization.

EROIC15.SetPixel

Allows writing a single pixel value in the ROI or image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetPixel(
    Euresys.Open_eVision_2_10.EC15 value,
    int x,
    int y
)
```

Parameters

value

-

x

-

y

-

Remarks

Although coordinates outside of the ROI can be supplied, this function will raise an error condition if the coordinates are outside of the top parent image. It should be noted that calling this function several thousand times can be slow. The recommended way to access pixel content in an image or ROI is to use the [EROIC15](#) and [EROIC15](#) functions. For BW8 images/ROIs, using the C++ API only, it is possible to access pixels data faster through an intermediate object: EBW8PixelAccessor.

EROIC15.TopParent

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EImageC15 TopParent
    { get; }
```

4.172. EROIC16 Class

The EROIC16 class is used to represent rectangular regions of interest inside C16 color images. See ROIs.

Base Class: [EBaseROI](#)

Derived Class(es): [EImageC16](#)

Namespace: Euresys.Open_eVision_2_10

Properties

[FirstSubROI](#)

See [GetFirstSubROI](#) in the derived classes

[NextSiblingROI](#)

This method returns the ROI that is the next sibling of this ROI.

[Parent](#)

-

[TopParent](#)

M-

e

Methods

[EROIC16](#)

-

[GetNextROI](#)

See [GetNextROI](#) in the derived classes

[GetPixel](#)

Allows reading a single pixel value in the ROI or image.

operator=

-

Serialize

Serializes the [EROIC16](#).

SetPixel

Allows writing a single pixel value in the ROI or image.

R

OIC16.EROIC16

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EROIC16 (
)
void EROIC16 (
    Euresys.Open_eVision_2_10.EROIC16 other
)
```

Parameters

other

-

EROIC16.FirstSubROI

See GetFirstSubROI in the derived classes

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
new Euresys.Open_eVision_2_10.EROIC16 FirstSubROI  
    { get; }
```

EROIC16.GetNextROI

See GetNextROI in the derived classes

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EROIC16 GetNextROI (  
    Euresys.Open_eVision_2_10.EBaseROI startROI  
)
```

Parameters

startROI

-

EROIC16.GetPixel

Allows reading a single pixel value in the ROI or image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EROIC16 GetPixel(
    int x,
    int y
)
```

Parameters

x
-
y
-

Remarks

Although coordinates outside of the ROI can be supplied, this function will raise an error condition if the coordinates are outside of the top parent image. It should be noted that calling this function several thousand times can be slow. The recommended way to access pixel content in an image or ROI is to use the [EROIC16](#) and [EROIC16](#) functions. For BW8 images/ROIs, using the C++ API only, it is possible to access pixels data faster through an intermediate object: EBW8PixelAccessor.

EROIC16.NextSiblingROI

This method returns the ROI that is the next sibling of this ROI.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
new Euresys.Open_eVision_2_10.EROIC16 NextSiblingROI
    { get; }
```

EROIC16.operator=

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EROIC16 operator=(  
    Euresys.Open_eVision_2_10.EROIC16 other  
)
```

Parameters

other

-

EROIC16.Parent

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
new Euresys.Open_eVision_2_10.EROIC16 Parent  
    { get; }
```

EROIC16.Serialize

Serializes the [EROIC16](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Serialize(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

Serializer used for the serialization.

EROIC16.SetPixel

Allows writing a single pixel value in the ROI or image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetPixel(
    Euresys.Open_eVision_2_10.EC16 value,
    int x,
    int y
)
```

Parameters

value

-

x

-

y

-

Remarks

Although coordinates outside of the ROI can be supplied, this function will raise an error condition if the coordinates are outside of the top parent image. It should be noted that calling this function several thousand times can be slow. The recommended way to access pixel content in an image or ROI is to use the [EROIC16](#) and [EROIC16](#) functions. For BW8 images/ROIs, using the C++ API only, it is possible to access pixels data faster through an intermediate object: `EBW8PixelAccessor`.

EROIC16.TopParent

-

Namespace: `Euresys.Open_eVision_2_10`

```
[C#]
Euresys.Open_eVision_2_10.EImageC16 TopParent
{ get; }
```

4.173. EROIC24 Class

The EROIC24 class is used to represent rectangular regions of interest inside C24 color images. See ROIs.

Base Class: [EBaseROI](#)

Derived Class(es): [EImageC24](#)

Namespace: `Euresys.Open_eVision_2_10`

Properties

[FirstSubROI](#)

See `GetFirstSubROI` in the derived classes

NextSiblingROI

This method returns the ROI that is the next sibling of this ROI.

Parent

-

TopParent

M-

e

thods

EROIC24

-

GetNextROI

See GetNextROI in the derived classes

GetPixel

Allows reading a single pixel value in the ROI or image.

operator=

-

Serialize

Serializes the [EROIC24](#).

SetPixel

E Allows writing a single pixel value in the ROI or image.

R

OIC24.EROIC24

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EROIC24 (
)
void EROIC24 (
    Euresys.Open_eVision_2_10.EROIC24 other
)
```

Parameters

other

-

EROIC24.FirstSubROI

See GetFirstSubROI in the derived classes

Namespace: Euresys.Open_eVision_2_10

```
[C#]
new Euresys.Open_eVision_2_10.EROIC24 FirstSubROI
{ get; }
```

EROIC24.GetNextROI

See GetNextROI in the derived classes

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EROIC24 GetNextROI (
    Euresys.Open_eVision_2_10.EBaseROI startROI
)
```

Parameters

startROI

-

EROIC24.GetPixel

Allows reading a single pixel value in the ROI or image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EC24 GetPixel (
    int x,
    int y
)
```

Parameters

x

-

y

-

Remarks

Although coordinates outside of the ROI can be supplied, this function will raise an error condition if the coordinates are outside of the top parent image. It should be noted that calling this function several thousand times can be slow. The recommended way to access pixel content in an image or ROI is to use the [EROIC24](#) and [EROIC24](#) functions. For BW8 images/ROIs, using the C++ API only, it is possible to access pixels data faster through an intermediate object: `EBW8PixelAccessor`.

EROIC24.NextSiblingROI

This method returns the ROI that is the next sibling of this ROI.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
new Euresys.Open_eVision_2_10.EROIC24 NextSiblingROI  
    { get; }
```

EROIC24.operator=

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EROIC24 operator=(  
    Euresys.Open_eVision_2_10.EROIC24 other  
)
```

Parameters

other

-

EROIC24.Parent

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
new Euresys.Open_eVision_2_10.EROIC24 Parent  
    { get; }
```

EROIC24.Serialize

Serializes the [EROIC24](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Serialize(  
    Euresys.Open_eVision_2_10.ESerializer serializer  
)
```

Parameters

serializer

Serializer used for the serialization.

EROIC24.SetPixel

Allows writing a single pixel value in the ROI or image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetPixel(
    Euresys.Open_eVision_2_10.EC24 value,
    int x,
    int y
)
```

Parameters

value

-

x

-

y

-

Remarks

Although coordinates outside of the ROI can be supplied, this function will raise an error condition if the coordinates are outside of the top parent image. It should be noted that calling this function several thousand times can be slow. The recommended way to access pixel content in an image or ROI is to use the [EROIC24](#) and [EROIC24](#) functions. For BW8 images/ROIs, using the C++ API only, it is possible to access pixels data faster through an intermediate object: [EBW8PixelAccessor](#).

EROIC24.TopParent

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EImageC24 TopParent
    { get; }
```

4.174. EROIC24A Class

The EROIC24A class is used to represent rectangular regions of interest inside C24A color images. See ROIs.

Base Class: [EBaseROI](#)

Derived Class(es): [EImageC24A](#)

Namespace: Euresys.Open_eVision_2_10

Properties

[FirstSubROI](#)

See [GetFirstSubROI](#) in the derived classes

[NextSiblingROI](#)

This method returns the ROI that is the next sibling of this ROI.

[Parent](#)

-

[TopParent](#)

M-

e

Methods

[EROIC24A](#)

-

[GetNextROI](#)

See [GetNextROI](#) in the derived classes

[GetPixel](#)

Allows reading a single pixel value in the ROI or image.

operator=

-

Serialize

Serializes the [EROIC24A](#).

SetPixel

Allows writing a single pixel value in the ROI or image.

R

EROIC24A.EROIC24A

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EROIC24A(
)
void EROIC24A(
    Euresys.Open_eVision_2_10.EROIC24A other
)
```

Parameters

other

-

EROIC24A.FirstSubROI

See GetFirstSubROI in the derived classes

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
new Euresys.Open_eVision_2_10.EROIC24A FirstSubROI  
    { get; }
```

EROIC24A.GetNextROI

See GetNextROI in the derived classes

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EROIC24A GetNextROI (  
    Euresys.Open_eVision_2_10.EBaseROI startROI  
)
```

Parameters

startROI

-

EROIC24A.GetPixel

Allows reading a single pixel value in the ROI or image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EROIC24A GetPixel(
    int x,
    int y
)
```

Parameters

x
-
y
-

Remarks

Although coordinates outside of the ROI can be supplied, this function will raise an error condition if the coordinates are outside of the top parent image. It should be noted that calling this function several thousand times can be slow. The recommended way to access pixel content in an image or ROI is to use the [EROIC24A](#) and [EROIC24A](#) functions. For BW8 images/ROIs, using the C++ API only, it is possible to access pixels data faster through an intermediate object: EBW8PixelAccessor.

EROIC24A.NextSiblingROI

This method returns the ROI that is the next sibling of this ROI.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
new Euresys.Open_eVision_2_10.EROIC24A NextSiblingROI
{ get; }
```

EROIC24A.operator=

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EROIC24A operator=(  
    Euresys.Open_eVision_2_10.EROIC24A other  
)
```

Parameters

other

EROIC24A.Parent

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
new Euresys.Open_eVision_2_10.EROIC24A Parent  
    { get; }
```

EROIC24A.Serialize

Serializes the [EROIC24A](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Serialize(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

Serializer used for the serialization.

EROIC24A.SetPixel

Allows writing a single pixel value in the ROI or image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetPixel(
    Euresys.Open_eVision_2_10.EC24A value,
    int x,
    int y
)
```

Parameters

value

-

x

-

y

-

Remarks

Although coordinates outside of the ROI can be supplied, this function will raise an error condition if the coordinates are outside of the top parent image. It should be noted that calling this function several thousand times can be slow. The recommended way to access pixel content in an image or ROI is to use the [EROIC24A](#) and [EROIC24A](#) functions. For BW8 images/ROIs, using the C++ API only, it is possible to access pixels data faster through an intermediate object: `EBW8PixelAccessor`.

EROIC24A.TopParent

-

Namespace: `Euresys.Open_eVision_2_10`

[C#]

```
Euresys.Open_eVision_2_10.EImageC24A TopParent  
{ get; }
```

4.175. EROIC48 Class

The EROIC48 class is used to represent rectangular regions of interest inside C48 color images. See ROIs.

Base Class: [EBaseROI](#)

Derived Class(es): [EImageC48](#)

Namespace: `Euresys.Open_eVision_2_10`

Properties

[FirstSubROI](#)

See `GetFirstSubROI` in the derived classes

NextSiblingROI

This method returns the ROI that is the next sibling of this ROI.

Parent

-

TopParent

M-

e

thods

EROIC48

-

GetNextROI

See GetNextROI in the derived classes

GetPixel

Allows reading a single pixel value in the ROI or image.

operator=

-

Serialize

Serializes the [EROIC48](#).

SetPixel

E Allows writing a single pixel value in the ROI or image.

R

OIC48.EROIC48

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EROIC48 (
)
void EROIC48 (
    Euresys.Open_eVision_2_10.EROIC48 other
)
```

Parameters

other

-

EROIC48.FirstSubROI

See GetFirstSubROI in the derived classes

Namespace: Euresys.Open_eVision_2_10

```
[C#]
new Euresys.Open_eVision_2_10.EROIC48 FirstSubROI
{ get; }
```

EROIC48.GetNextROI

See GetNextROI in the derived classes

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EROIC48 GetNextROI (
    Euresys.Open_eVision_2_10.EBaseROI startROI
)
```

Parameters

startROI

-

EROIC48.GetPixel

Allows reading a single pixel value in the ROI or image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EC48 GetPixel (
    int x,
    int y
)
```

Parameters

x

-

y

-

Remarks

Although coordinates outside of the ROI can be supplied, this function will raise an error condition if the coordinates are outside of the top parent image. It should be noted that calling this function several thousand times can be slow. The recommended way to access pixel content in an image or ROI is to use the [EROIC48](#) and [EROIC48](#) functions. For BW8 images/ROIs, using the C++ API only, it is possible to access pixels data faster through an intermediate object: [EBW8PixelAccessor](#).

EROIC48.NextSiblingROI

This method returns the ROI that is the next sibling of this ROI.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
new Euresys.Open_eVision_2_10.EROIC48 NextSiblingROI  
    { get; }
```

EROIC48.operator=

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EROIC48 operator=(  
    Euresys.Open_eVision_2_10.EROIC48 other  
)
```

Parameters

other

-

EROIC48.Parent

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
new Euresys.Open_eVision_2_10.EROIC48 Parent
    { get; }
```

EROIC48.Serialize

Serializes the [EROIC48](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Serialize(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

Serializer used for the serialization.

EROIC48.SetPixel

Allows writing a single pixel value in the ROI or image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetPixel(
    Euresys.Open_eVision_2_10.EC48 value,
    int x,
    int y
)
```

Parameters

value

-

x

-

y

-

Remarks

Although coordinates outside of the ROI can be supplied, this function will raise an error condition if the coordinates are outside of the top parent image. It should be noted that calling this function several thousand times can be slow. The recommended way to access pixel content in an image or ROI is to use the [EROIC48](#) and [EROIC48](#) functions. For BW8 images/ROIs, using the C++ API only, it is possible to access pixels data faster through an intermediate object: EBW8PixelAccessor.

EROIC48.TopParent

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EImageC48 TopParent
{ get; }
```


4.176. ERotatedBoundingBox Class

This class represents a rotated bounding box.

Remarks

The rotated bounding box is a rotated, rectangular surface. Its coordinates are floating-point, which makes this class appropriate to handle sub-pixel surfaces.

Namespace: Euresys.Open_eVision_2_10

Properties

Angle	Returns the angle of the bounding box (in the current angle units).
Center	Returns the coordinate of the center of the bounding box.
CenterX	Returns the abscissa of the center of the bounding box.
CenterY	Returns the ordinate of the center of the bounding box.
Height	Returns the height of the bounding box.
Quadrangle	Returns the coordinates of the four corners of the bounding box.
Width	Returns the width of the bounding box.

Methods

Draw	Draws the rotated bounding box.
DrawWithCurrentPen	Draws the rotated bounding box.
ERotatedBoundingBox	Constructor of the rotated bounding box.
LocalToGlobalBox	Transforms a (local) rotated bounding box, as another (global) rotated bounding box whose coordinates are defined relatively to the current rotated bounding box.
LocalToGlobalPoint	Transforms a (local) point, as another (global) point whose coordinates are defined relatively to the current rotated bounding box.
operator=	-
Translate	Applies a translation on the center of the bounding box.

R

rotatedBoundingBox.Angle

Returns the angle of the bounding box (in the current angle units).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float Angle
```

```
{ get; }
```

ERotatedBoundingBox.Center

Returns the coordinate of the center of the bounding box.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EPoint Center
```

```
{ get; }
```

ERotatedBoundingBox.CenterX

Returns the abscissa of the center of the bounding box.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float CenterX
```

```
{ get; }
```

ERotatedBoundingBox.CenterY

Returns the ordinate of the center of the bounding box.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float CenterY  
    { get; }
```

ERotatedBoundingBox.Draw

Draws the rotated bounding box.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Draw(  
    IntPtr graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    bool drawDiagonals  
)
```

```

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    bool drawDiagonals
)

void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    bool drawDiagonals
)

```

Parameters

graphicContext

Graphic context on which to draw.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

drawDiagonals

Specifies whether or not the diagonals of the bounding box are drawn.

color

The color in which to draw the overlay.

Remarks

Drawing is done in the device context associated to the desired window.

ERotatedBoundingBox.DrawWithCurrentPen

Draws the rotated bounding box.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawWithCurrentPen(
    IntPtr graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    bool drawDiagonals
)
```

Parameters

graphicContext

Graphic context on which to draw.

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0** (default), the horizontal zooming factor is used instead, so as to provide isotropic zooming.

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

drawDiagonals

Specifies whether or not the diagonals of the bounding box are drawn.

Remarks

Drawing is done in the device context associated to the desired window.

ERotatedBoundingBox.ERotatedBoundingBox

Constructor of the rotated bounding box.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ERotatedBoundingBox (
    float centerX,
    float centerY,
    float width,
    float height,
    float angle
)

void ERotatedBoundingBox (
)

void ERotatedBoundingBox (
    Euresys.Open_eVision_2_10.ERotatedBoundingBox other
)
```

Parameters

centerX

The abscissa of the center of the bounding box.

centerY

The ordinate of the center of the bounding box.

width

The width of the bounding box.

height

The height of the bounding box.

angle

The angle of the bounding box (in the current angle units).

other

-

ERotatedBoundingBox.Height

Returns the height of the bounding box.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Height  
    { get; }
```

ERotatedBoundingBox.LocalToGlobalBox

Transforms a (local) rotated bounding box, as another (global) rotated bounding box whose coordinates are defined relatively to the current rotated bounding box.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.ERotatedBoundingBox LocalToGlobalBox(  
    Euresys.Open_eVision_2_10.ERotatedBoundingBox localBox  
)
```

Parameters

localBox

-

ERotatedBoundingBox.LocalToGlobalPoint

Transforms a (local) point, as another (global) point whose coordinates are defined relatively to the current rotated bounding box.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint LocalToGlobalPoint(
    Euresys.Open_eVision_2_10.EPoint localPoint
)
```

Parameters

localPoint

-

ERotatedBoundingBox.operator=

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ERotatedBoundingBox operator=(
    Euresys.Open_eVision_2_10.ERotatedBoundingBox other
)
```

Parameters

other

-

ERotatedBoundingBox.Quadrangle

Returns the coordinates of the four corners of the bounding box.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EQuadrangle Quadrangle
{ get; }
```

ERotatedBoundingBox.Translate

Applies a translation on the center of the bounding box.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Translate(
    float offsetX,
    float offsetY
)
```

Parameters

offsetX

The offset along the X-axis.

offsetY

The offset along the Y-axis.

ERotatedBoundingBox.Width

Returns the width of the bounding box.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Width  
    { get; }
```

4.177. ESamplePoint Class

A point sampled by a gauge.

Namespace: Euresys.Open_eVision_2_10

Properties

IsOutlier

Outlier status of the sample point

IsValid

Validity of the sample point

Position

Position of the sample point

Methods

ESamplePoint

Constructor

operator=

ESamplePoint.ESamplePoint

Copies all the data from another ESamplePoint object into the current

Constructor

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ESamplePoint(
)
void ESamplePoint(
    Euresys.Open_eVision_2_10.ESamplePoint other
)
```

Parameters

other

-

ESamplePoint.IsOutlier

Outlier status of the sample point

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
bool IsOutlier
{ get; }
```

ESamplePoint.IsValid

Validity of the sample point

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool IsValid
{ get; }
```

ESamplePoint.operator=

Copies all the data from another ESamplePoint object into the current ESamplePoint object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ESamplePoint operator=(
    Euresys.Open_eVision_2_10.ESamplePoint other
)
```

Parameters

other

ESamplePoint object to be copied.

ESamplePoint.Position

Position of the sample point

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint Position  
{ get; }
```

4.178. EScaleCalibrationModel Class

[EScaleCalibrationModel](#) is used to convert depth map 2.5D point to 3D world position, only by applying a scale factor.

That kind of "calibration" does not correct the perspective or distortion present in depth maps.

Is a simple and fast way to get a 3D point cloud by applying a scale to each coordinate axis of the depth map pixels.

Base Class: [ECalibrationModel](#)

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

[FactorX](#)

Returns the width of a pixel in metric unit.

[FactorY](#)

Returns the distance between 2 profiles (depth map lines) in metric unit.

FactorZ

Returns the scale of the pixel value in metric unit.

Type

M Returns the type of calibration model, see [ECalibrationType](#).

e

Methods

[EScaleCalibrationModel](#)

Constructs a [EScaleCalibrationModel](#). Parameters are initialized to default unit values.

Load

Loads the [EScaleCalibrationModel](#) calibration model. The given [ESerializer](#) must have been created for reading.

operator=

Assignment operator.

operator==

Comparison operator.

Save

E Saves the [EScaleCalibrationModel](#) calibration model. The given [ESerializer](#) must have been created for writing.

S

caleCalibrationModel.EScaleCalibrationModel

Constructs a [EScaleCalibrationModel](#). Parameters are initialized to default unit values.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void EScaleCalibrationModel (
)

void EScaleCalibrationModel (
    float factorX,
    float factorY,
    float factorZ
)

void EScaleCalibrationModel (
    Euresys.Open_eVision_2_10.Easy3D.EScaleCalibrationModel other
)
```

Parameters

factorX

Width of a pixel in metric unit (factor for X coordinate).

factorY

Distance between 2 profiles (depth map lines) in metric unit (factor for Y coordinate).

factorZ

Scale of the pixel value in metric unit (factor for Z coordinate).

other

Another [EScaleCalibrationModel](#) used for the initialization.

EScaleCalibrationModel.FactorX

Returns the width of a pixel in metric unit.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
float FactorX
{ get; }
```


EScaleCalibrationModel.FactoryY

Returns the distance between 2 profiles (depth map lines) in metric unit.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float FactoryY  
    { get; }
```

EScaleCalibrationModel.FactoryZ

Returns the scale of the pixel value in metric unit.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
float FactoryZ  
    { get; }
```

EScaleCalibrationModel.Load

Loads the [EScaleCalibrationModel](#) calibration model. The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Load(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The serializer.

EScaleCalibrationModel.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EScaleCalibrationModel operator=(
    Euresys.Open_eVision_2_10.Easy3D.EScaleCalibrationModel other
)
```

Parameters

other

Another [EScaleCalibrationModel](#).

EScaleCalibrationModel.operator==

Comparison operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool operator==(
    Euresys.Open_eVision_2_10.Easy3D.EScaleCalibrationModel other
)
```

Parameters

other

Another [EScaleCalibrationModel](#).

EScaleCalibrationModel.Save

Saves the [EScaleCalibrationModel](#) calibration model. The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The serializer.

EScaleCalibrationModel.Type

Returns the type of calibration model, see [ECalibrationType](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override Euresys.Open_eVision_2_10.Easy3D.ECalibrationType Type
{ get; }
```

4.179. ESearchParamsType Class

This class is instantiated once in each [EMatrixCodeReader](#) and represents the search parameters that are explored when reading a [EMatrixCode](#).

Remarks

This class contains 4 sets of search parameters that are scanned at read time. At [EMatrixCodeReader](#) construction time, these sets of values are initialized with all possible values. This means, for instance, that a data matrix code read in a freshly created reader is matched against all possible logical sizes. As a consequence, the default values of these sets are not repeated here. They are assumed to represent every possible search parameters. The [ESearchParamsType](#) object needs to be used only if you wish to "override" the learning process.

Namespace: Euresys.Open_eVision_2_10

Properties

ContrastCount

The size of the list of [EMatrixCodeContrastMode](#) the candidate is matched against at read time.

FamilyCount

The size of the list of [EFamily](#) the candidate is matched against at read time.

FlippingCount

The size of the list of [EFlipping](#) the candidate is matched against at read time.

LogicalSizeCount

M The size of the list of [ELogicalSize](#) the candidate is matched against at read time.

e

thods

AddContrast

Adds a new contrast mode to the list of [EMatrixCodeContrastMode](#) the candidate is matched against at read time.

AddFamily

Adds a new family to the list of [EFamily](#) the candidate is matched against at read time.

AddFlipping

Adds a new flipping to the list of [EFlipping](#) the candidate is matched against at read time.

AddLogicalSize

Adds a new logical size to the list of [ELogicalSize](#) the candidate is matched against at read time.

ClearContrast

Clears the list of contrast modes the candidate is matched against at read time.

ClearFamily

Clears the list of families the candidate is matched against at read time.

ClearFlipping

Clears the list of flippings the candidate is matched against at read time.

ClearLogicalSize

Clears the list of logical sizes the candidate is matched against at read time.

GetContrast

Gets an item in the list of [EMatrixCodeContrastMode](#) the candidate is matched against at read time.

GetFamily

Gets an item in the list of [EFamily](#) the candidate is matched against at read time.

GetFlipping

Gets an item in the list of [EFlipping](#) the candidate is matched against at read time.

GetLogicalSize

Gets an item in the list of [ELogicalSize](#) the candidate is matched against at read time.

RemoveContrast

Removes the contrast mode from the list of [EMatrixCodeContrastMode](#) the candidate is matched against at read time.

RemoveFamily

Removes the family from the list of [EFamily](#) the candidate is matched against at read time.

RemoveFlipping

Removes the flipping from the list of [EFlipping](#) the candidate is matched against at read time.

RemoveLogicalSize

Removes the logical size from the list of [ELogicalSize](#) the candidate is matched against at read time.

ESearchParamsType.AddContrast

Adds a new contrast mode to the list of [EMatrixCodeContrastMode](#) the candidate is matched against at read time.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddContrast(
    Euresys.Open_eVision_2_10.EMatrixCodeContrastMode searchContrast
)
```

Parameters

searchContrast

Contrast mode to add to the list.

ESearchParamsType.AddFamily

Adds a new family to the list of [EFamily](#) the candidate is matched against at read time.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddFamily(
    Euresys.Open_eVision_2_10.EFamily searchFamily
)
```

Parameters

searchFamily

Family to add to the list.

ESearchParamsType.AddFlipping

Adds a new flipping to the list of [EFlipping](#) the candidate is matched against at read time.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddFlipping(
    Euresys.Open_eVision_2_10.EFlipping searchFlipping
)
```

Parameters

searchFlipping

Flipping to add to the list.

ESearchParamsType.AddLogicalSize

Adds a new logical size to the list of [ELogicalSize](#) the candidate is matched against at read time.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddLogicalSize(
    Euresys.Open_eVision_2_10.ELogicalSize searchLogicalSize
)
```

Parameters

searchLogicalSize

Logical size to add to the list.

ESearchParamsType.ClearContrast

Clears the list of contrast modes the candidate is matched against at read time.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void ClearContrast(  
)
```

ESearchParamsType.ClearFamily

Clears the list of families the candidate is matched against at read time.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void ClearFamily(  
)
```

ESearchParamsType.ClearFlipping

Clears the list of flippings the candidate is matched against at read time.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ClearFlipping(
)
```

ESearchParamsType.ClearLogicalSize

Clears the list of logical sizes the candidate is matched against at read time.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void ClearLogicalSize(
)
```

ESearchParamsType.ContrastCount

The size of the list of [EMatrixCodeContrastMode](#) the candidate is matched against at read time.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int ContrastCount
{ get; }
```

ESearchParamsType.FamilyCount

The size of the list of [EFamily](#) the candidate is matched against at read time.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int FamilyCount  
    { get; }
```

ESearchParamsType.FlippingCount

The size of the list of [EFlipping](#) the candidate is matched against at read time.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int FlippingCount  
    { get; }
```

ESearchParamsType.GetContrast

Gets an item in the list of [EMatrixCodeContrastMode](#) the candidate is matched against at read time.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EMatrixCodeContrastMode GetContrast(
    int index
)
```

Parameters

index

Position in the list.

ESearchParamsType.GetFamily

Gets an item in the list of [EFamily](#) the candidate is matched against at read time.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EFamily GetFamily(
    int index
)
```

Parameters

index

Position in the list.

ESearchParamsType.GetFlipping

Gets an item in the list of [EFlipping](#) the candidate is matched against at read time.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EFlipping GetFlipping(
    int index
)
```

Parameters

index

Position in the list.

ESearchParamsType.GetLogicalSize

Gets an item in the list of [ELogicalSize](#) the candidate is matched against at read time.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ELogicalSize GetLogicalSize(
    int index
)
```

Parameters

index

Position in the list.

ESearchParamsType.LogicalSizeCount

The size of the list of [ELogicalSize](#) the candidate is matched against at read time.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int LogicalSizeCount
{ get; }
```

ESearchParamsType.RemoveContrast

Removes the contrast mode from the list of [EMatrixCodeContrastMode](#) the candidate is matched against at read time.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void RemoveContrast(
    Euresys.Open_eVision_2_10.EMatrixCodeContrastMode searchContrast
)
```

Parameters

searchContrast

Contrast mode to remove from the list.

ESearchParamsType.RemoveFamily

Removes the family from the list of [EFamily](#) the candidate is matched against at read time.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void RemoveFamily(  
    Euresys.Open_eVision_2_10.EFamily searchFamily  
)
```

Parameters

searchFamily

Family to remove from the list.

ESearchParamsType.RemoveFlipping

Removes the flipping from the list of [EFlipping](#) the candidate is matched against at read time.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void RemoveFlipping(  
    Euresys.Open_eVision_2_10.EFlipping searchFlipping  
)
```

Parameters

searchFlipping

Flipping to remove from the list.

ESearchParamsType.RemoveLogicalSize

Removes the logical size from the list of [ELogicalSize](#) the candidate is matched against at read time.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void RemoveLogicalSize(
    Euresys.Open_eVision_2_10.ELogicalSize searchLogicalSize
)
```

Parameters

searchLogicalSize

Logical size to remove from the list.

4.180. ESerializer Class

Abstract interface for file-like objects.

Remarks

The ESerializer object manages operations of reading from and writing to an archive (a file on the system hard disk, for instance). ESerializer objects cannot be instantiated directly. To create an ESerializer object, one of the following static factory methods has to be used:

Note. An ESerializer object can not be used in the same time for reading and writing. So, [ESerializer::CreateFileWriter](#) creates an ESerializer object that should be used with **Save** methods and [ESerializer::CreateFileReader](#) creates an ESerializer object that should be used with **Load** methods.

Derived Class(es): [EFilePointerSerializer](#) [EFileSerializer](#) [EMemorySerializer](#)

Namespace: Euresys.Open_eVision_2_10

Properties

Writing

M Returns **TRUE** if the [ESerializer](#) object has been created for writing and **FALSE** otherwise.

e

Methods

Close

Closes the file associated with the [ESerializer](#) object.

CreateFileReader

Returns an ESerializer object suitable for reading from a file.

CreateFileWriter

Returns an ESerializer object suitable for opening a file and writing into it.

CreateMemoryReader

Returns an ESerializer object suitable for reading from a buffer.

CreateMemoryWriter

Returns an ESerializer object suitable for serializing into a buffer.

S

erializer.Close

Closes the file associated with the [ESerializer](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Close(  
)
```

ESerializer.CreateFileReader

Returns an ESerializer object suitable for reading from a file.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ESerializer CreateFileReader(
    string filePath
)
```

Parameters

filePath

Full path and name specification of the file to be used to create the ESerializer object.

Remarks

It is up to users to delete the ESerializer object when they have done using it in **Load** calls. If the call does not succeed, it returns **NULL**. Please check the Open eVision error code to get further informations.

ESerializer.CreateFileWriter

Returns an ESerializer object suitable for opening a file and writing into it.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ESerializer CreateFileWriter(
    string filePath,
    Euresys.Open_eVision_2_10.ESerializerFileWriterMode mode
)
```

Parameters

filePath

Full path and name specification of the file to be used to create the ESerializer object.

mode

Creation mode of the storage file, as defined by [ESerializerFileWriterMode](#) (by default, **Create**).

Remarks

The `ESerializerFileWriterMode` parameter is an enumerated type that allows to control what happens when the file already exists: * If **mode** is **Create**, the call will not succeed if the file already exists. * If **mode** is **Overwrite**, the existing file will be overwritten. * If **mode** is **Append**, the new data will be appended to the existing file content. It is up to users to delete the `ESerializer` object when they have done using it in **Save** calls. If the call does not succeed, it returns **NULL**. Please check the Open eVision error code to get further information.

ESerializer.CreateMemoryReader

Returns an `ESerializer` object suitable for reading from a buffer.

Namespace: `Euresys.Open_eVision_2_10`

```
[C#]
Euresys.Open_eVision_2_10.ESerializer CreateMemoryReader(
    ref byte buffer,
    uint size
)
```

Parameters

buffer

Address of the buffer to be used by the memory serializer.

size

Size of the buffer to be used by the memory serializer.

Remarks

The buffer is copied inside the internal memory of the serializer. It is up to users to delete the `ESerializer` object when they have done using it in **Load** calls.

ESerializer.CreateMemoryWriter

Returns an `ESerializer` object suitable for serializing into a buffer.

Namespace: `Euresys.Open_eVision_2_10`

```
[C#]
Euresys.Open_eVision_2_10.ESerializer CreateMemoryWriter(
)
Euresys.Open_eVision_2_10.ESerializer CreateMemoryWriter(
    uint initialBufferSize
)
Euresys.Open_eVision_2_10.ESerializer CreateMemoryWriter(
    ref byte buffer,
    uint size
)
```

Parameters

initialBufferSize

-

buffer

Address of the buffer to be used by the memory serializer.

size

Size of the buffer to be used by the memory serializer.

Remarks

If a buffer is not provided, a buffer will be automatically created. Please note that, in this case, if you didn't provide a size, the buffer will be automatically resized as needed. It is up to users to delete the ESerializer object when they have done using it in **Save** calls. The returned serializer underlying type is [EMemorySerializer](#), for more information, see the class documentation.

ESerializer.Writing

Returns **TRUE** if the [ESerializer](#) object has been created for writing and **FALSE** otherwise.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
abstract bool Writing
```

```
{ get; }
```

4.181. EShape Class

Abstract class to federate the classes that can be hierarchically attached together (from a geometrical point of view).

Derived Class(es): [EWorldShape](#) [ERectangleShape](#) [ECircleShape](#) [EFrameShape](#) [ELineShape](#) [EPointShape](#) [EWedgeShape](#)

Namespace: Euresys.Open_eVision_2_10

Properties

Active

Flag indicating whether the shape is active or not.

ActiveRecursive

Sets the flag indicating whether the shape and all its daughters are active or not.

ActualShape

Flag indicating whether an inquiry returns a result pertaining to the nominal gauge (**FALSE**, default) or the fitted model (**TRUE**).

ActualShapeRecursive

Sets the flag indicating whether an inquiry returns a result pertaining to the nominal gauge (**FALSE**, default) or the fitted model (**TRUE**). The flag is set in this shape and all its daughters.

ClosestShape

Closest shape among the daughters.

Dragable

Flag indicating whether the shape can be dragged or not.

DragableRecursive	Sets the flag indicating whether the shape and all its daughters can be dragged or not.
HitHandle	Handle currently under the cursor.
HitShape	Pointer to the shape currently under the cursor.
Labeled	Flag indicating whether the shape label should be displayed or not.
LabeledRecursive	Sets the flag indicating whether the shape label should be displayed or not. The flag is set in this shape and all its daughters.
Mother	Pointer to the mother shape.
Name	Name of the EShape object.
NumDaughters	Number of daughters attached to the shape.
PanX	Current horizontal panning factor for drawing operations.
PanY	Current vertical panning factor for drawing operations.
Resizable	Flag indicating whether the shape can be resized or not.

ResizableRecursive

Sets the flag indicating whether the shape and all its daughters can be resized or not.

Rotatable

Flag indicating whether the shape can be rotated or not.

RotatableRecursive

Sets the flag indicating whether the shape and all its daughters can be rotated or not.

Selectable

Flag indicating whether the shape can be selected or not.

SelectableRecursive

Sets the flag indicating whether the shape and all its daughters can be selected or not.

Selected

Flag indicating whether the shape is selected or not.

SelectedRecursive

Sets the flag indicating whether the shape and all its daughters are selected or not.

Type

Shape type.

Visible

Flag indicating whether the shape is visible or not.

VisibleRecursive

Sets the flag indicating whether the shape and its daughter shapes are visible or not.

WorldShape

World ancestor.

ZoomX

Current horizontal zooming factor for drawing operations.

ZoomY

M Current vertical zooming factor for drawing operations.

e

thods

Attach

Attaches the gauge to a mother gauge or shape.

Closest

Find the daughter shape that is the closest to this shape. To retrieve the closest shape, use [EShape::ClosestShape](#).

Detach

Detaches the gauge from its mother gauge or shape.

DetachDaughters

Detaches the daughter gauges or shapes.

DisableBehaviorFilter

Disables (i.e. removes) a condition from the list of conditions in the behavior filter.

DisableTypeFilter

Enables all shape types

Drag

Moves a handle to a new position and updates the position parameters of the shape.

Draw	Draws a graphical representation of a shape, as defined by EDrawingMode .
DrawWithCurrentPen	Draws a graphical representation of a shape, as defined by EDrawingMode .
EnableBehaviorFilter	Modifies the so-called behavior filter that is a conjunction of conditions specifying when a shape or a gauge should be displayed or taken into consideration when monitoring the mouse interactions.
EnableTypeFilter	Enables the filter of the specified shape type
GetAllocated	Gets the allocated flag.
GetDaughter	Returns a pointer to the specified daughter gauge or shape.
GetDraggingMode	Gets the dragging mode which defines how the shape could be dragged.
GetShapeNamed	Returns a pointer to a daughter gauge or shape specified by its name.
HitTest	Checks if there is a handle under the cursor.

InvalidateWorld

Invalidates the world shape for this shape and all its ancestors

Load

Loads a classifier. The given [ESerializer](#) must have been created for reading.

LocalToSensor

Transforms a point from local coordinates to sensor coordinates.

Save

Loads a classifier. The given [ESerializer](#) must have been created for writing.

SensorToLocal

Transforms a point from sensor coordinates to local coordinates.

SetAllocated

Sets the allocated flag.

SetCursor

Sets the cursor current coordinates.

SetDraggingMode

Sets the dragging mode which defines how the shape could be dragged.

SetPan

Sets the horizontal and vertical panning factors for drawing operations.

SetZoom

Sets the horizontal and vertical zooming factors for drawing operations.

EShape.Active

Flag indicating whether the shape is active or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
virtual bool Active  
  
{ get; set; }
```

EShape.ActiveRecursive

Sets the flag indicating whether the shape and all its daughters are active or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
virtual bool ActiveRecursive  
  
{ get; set; }
```

Remarks

When complex gauging is required, several gauges can be grouped together. Applying [EShape::Process](#) to the mother gauge or shape triggers the measurement of the whole. Only the active gauges will participate in the process.

EShape.ActualShape

Flag indicating whether an inquiry returns a result pertaining to the nominal gauge (**FALSE**, default) or the fitted model (**TRUE**).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
bool ActualShape  
  
    { get; set; }
```

EShape.ActualShapeRecursive

Sets the flag indicating whether an inquiry returns a result pertaining to the nominal gauge (**FALSE**, default) or the fitted model (**TRUE**). The flag is set in this shape and all its daughters.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
bool ActualShapeRecursive  
  
    { get; set; }
```

EShape.Attach

Attaches the gauge to a mother gauge or shape.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void Attach(  
    Euresys.Open_eVision_2_10.EShape mother  
)
```

Parameters

mother

Pointer to the mother gauge or shape.

Remarks

When attached to a mother gauge, be aware that daughter gauges are not positioned according to the nominal mother gauge position, but to its corresponding fitted model.

EShape.Closest

Find the daughter shape that is the closest to this shape. To retrieve the closest shape, use [EShape::ClosestShape](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Closest(  
)
```

EShape.ClosestShape

Closest shape among the daughters.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EShape ClosestShape  
{ get; }
```

Remarks

Use [EShape::Closest](#) to recompute the closest shape.

EShape.Detach

Detaches the gauge from its mother gauge or shape.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Detach(  
)
```

EShape.DetachDaughters

Detaches the daughter gauges or shapes.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void DetachDaughters(  
)
```

EShape.DisableBehaviorFilter

Disables (i.e. removes) a condition from the list of conditions in the behavior filter.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DisableBehaviorFilter(
    Euresys.Open_eVision_2_10.EShapeBehavior behavior
)
```

Parameters

behavior

The behavior of the shape to be removed from the behavior filter.

Remarks

The condition to be disabled is identified by the behavior about which the condition is. Disabling a behavior leads to less restrictive conditions for the **Draw** and **HitTest** methods to be actually carried on.

EShape.DisableTypeFilter

Enables all shape types

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DisableTypeFilter(
)
```

EShape.Drag

Moves a handle to a new position and updates the position parameters of the shape.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Drag(
    int n32CursorX,
    int n32CursorY
)
```

Parameters

n32CursorX

Current cursor coordinates.

n32CursorY

Current cursor coordinates.

EShape.Dragable

Flag indicating whether the shape can be dragged or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool Dragable
{ get; set; }
```

EShape.DragableRecursive

Sets the flag indicating whether the shape and all its daughters can be dragged or not.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
bool DraggableRecursive
{ get; set; }
```

EShape.Draw

Draws a graphical representation of a shape, as defined by [EDrawingMode](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)

void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingMode

Indicates how the point location or model fitting gauge must be displayed, as defined by [EDrawingMode](#).

daughters

TRUE if the daughters gauges are to be displayed also.

color

-

EShape.DrawWithCurrentPen

Draws a graphical representation of a shape, as defined by [EDrawingMode](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawWithCurrentPen (
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingMode

Indicates how the point location or model fitting gauge must be displayed, as defined by [EDrawingMode](#).

daughters

TRUE if the daughters gauges are to be displayed also.

EShape.EnableBehaviorFilter

Modifies the so-called **behavior filter** that is a conjunction of conditions specifying when a shape or a gauge should be displayed or taken into consideration when monitoring the mouse interactions.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EnableBehaviorFilter(
    Euresys.Open_eVision_2_10.EShapeBehavior behavior,
    bool value
)
```

Parameters

behavior

The behavior of the shape to be tested.

value

The value at which the behavior property should be set to pass the test. By default, equals **True**.

Remarks

This method registers a new necessary condition for the **Draw** and **HitTest** families of methods to be actually carried on. Such a condition is about the behavior of the shape, as specified by the **behavior** argument. Initially, the behavior filter contains an empty list of conditions, which means that the **Draw** and **HitTest** methods will always be executed. Adding a new condition through [EShape::EnableBehaviorFilter](#) will introduce a new restriction on the effective execution of these methods. Use [EShape::DisableBehaviorFilter](#) to remove a condition from the behavior filter.

EShape.EnableTypeFilter

Enables the filter of the specified shape type

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EnableTypeFilter(
    uint un32Types
)
```

Parameters

un32Types

The type of the shape to filter.

EShape.GetAllocated

Gets the allocated flag.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool GetAllocated(  
    )
```

EShape.GetDaughter

Returns a pointer to the specified daughter gauge or shape.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EShape GetDaughter(  
    uint index  
    )
```

Parameters

index

Daughter gauge or shape index.

EShape.GetDraggingMode

Gets the dragging mode which defines how the shape could be dragged.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EDraggingMode GetDraggingMode(  
    )
```

EShape.GetShapeNamed

Returns a pointer to a daughter gauge or shape specified by its name.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EShape GetShapeNamed(  
    string name  
    )
```

Parameters

name

Name of the daughter gauge or shape.

EShape.HitHandle

Handle currently under the cursor.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EDragHandle HitHandle  
  
{ get; }
```

Remarks

When the cursor is over a particular handle, its shape could be changed for feedback.

EShape.HitShape

Pointer to the shape currently under the cursor.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EShape HitShape  
  
{ get; }
```

Remarks

When the cursor is over a particular shape, its shape could be changed for feedback.

EShape.HitTest

Checks if there is a handle under the cursor.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
bool HitTest(  
    bool bDaughters  
)
```

Parameters

bDaughters

Indicates if the check must be done in the whole hierarchy or just this object.

EShape.InvalidateWorld

Invalidates the world shape for this shape and all its ancestors

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void InvalidateWorld(  
)
```

EShape.Labeled

Flag indicating whether the shape label should be displayed or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool Labeled  
{ get; set; }
```

EShape.LabeledRecursive

Sets the flag indicating whether the shape label should be displayed or not. The flag is set in this shape and all its daughters.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool LabeledRecursive
    { get; set; }
```

EShape.Load

Loads a classifier. The given [ESerializer](#) must have been created for reading.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Load(
    Euresys.Open_eVision_2_10.ESerializer serializer,
    bool daughters
)
```

Parameters

serializer

Pointer to the [ESerializer](#) created for reading.

daughters

Indicates if the load must be done on the whole hierarchy or just this object.

EShape.LocalToSensor

Transforms a point from local coordinates to sensor coordinates.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint LocalToSensor(  
    Euresys.Open_eVision_2_10.EPoint LPoint  
)
```

Parameters

LPoint

The point in local coordinates.

EShape.Mother

Pointer to the mother shape.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EShape Mother  
    { get; }
```

EShape.Name

Name of the [EShape](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
string Name  
    { get; set; }
```

EShape.NumDaughters

Number of daughters attached to the shape.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint NumDaughters  
    { get; }
```

EShape.PanX

Current horizontal panning factor for drawing operations.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
virtual float PanX  
    { get; }
```

EShape.PanY

Current vertical panning factor for drawing operations.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
virtual float PanY  
  
    { get; }
```

EShape.Resizable

Flag indicating whether the shape can be resized or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool Resizable  
  
    { get; set; }
```

EShape.ResizableRecursive

Sets the flag indicating whether the shape and all its daughters can be resized or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
bool ResizableRecursive
```

```
{ get; set; }
```

EShape.Rotatable

Flag indicating whether the shape can be rotated or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
bool Rotatable
```

```
{ get; set; }
```

EShape.RotatableRecursive

Sets the flag indicating whether the shape and all its daughters can be rotated or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
bool RotatableRecursive
```

```
{ get; set; }
```

EShape.Save

Loads a classifier. The given [ESerializer](#) must have been created for writing.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Save(
    Euresys.Open_eVision_2_10.ESerializer serializer,
    bool daughters
)
```

Parameters

serializer

Pointer to the [ESerializer](#) created for writing.

daughters

Indicates if the save must be done on the whole hierarchy or just this object.

EShape.Selectable

Flag indicating whether the shape can be selected or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool Selectable
    { get; set; }
```

EShape.SelectableRecursive

Sets the flag indicating whether the shape and all its daughters can be selected or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool SelectableRecursive  
    { get; set; }
```

EShape.Selected

Flag indicating whether the shape is selected or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool Selected  
    { get; set; }
```

EShape.SelectedRecursive

Sets the flag indicating whether the shape and all its daughters are selected or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
bool SelectedRecursive  
  
{ get; set; }
```

EShape.SensorToLocal

Transforms a point from sensor coordinates to local coordinates.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EPoint SensorToLocal(  
    Euresys.Open_eVision_2_10.EPoint SPoint  
)
```

Parameters

SPoint

The point in sensor coordinates.

EShape.SetAllocated

Sets the allocated flag.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void SetAllocated(  
    bool bAllocated,  
    bool bDaughters  
)
```

Parameters

bAllocated

Whether to set the allocated flag or not.

bDaughters

Indicates if the set must be done in the whole hierarchy or just this object.

EShape.SetCursor

Sets the cursor current coordinates.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void SetCursor(  
    int x,  
    int y  
)
```

Parameters

x

Cursor current coordinates.

y

Cursor current coordinates.

EShape.SetDraggingMode

Sets the dragging mode which defines how the shape could be dragged.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetDraggingMode(
    Euresys.Open_eVision_2_10.EDraggingMode eDraggingMode,
    bool bDaughters
)
```

Parameters

eDraggingMode

The draggingMode.

bDaughters

Indicates if the set must be done in the whole hierarchy or just this object.

EShape.SetPan

Sets the horizontal and vertical panning factors for drawing operations.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetPan(
    float f32PanX,
    float f32PanY
)
```

Parameters

f32PanX

-
f32PanY

Remarks

All objects attached to an [EWorldShape](#) object inherit of the same panning factor.

EShape.SetZoom

Sets the horizontal and vertical zooming factors for drawing operations.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void SetZoom(  
    float f32ZoomX,  
    float f32ZoomY  
)
```

Parameters

f32ZoomX

-

f32ZoomY

-

Remarks

All objects attached to an [EWorldShape](#) inherit of the same zooming factor.

EShape.Type

Shape type.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
abstract Euresys.Open_eVision_2_10.EShapeType Type  
  
    { get; }
```

EShape.Visible

Flag indicating whether the shape is visible or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool Visible  
  
    { get; set; }
```

EShape.VisibleRecursive

Sets the flag indicating whether the shape and its daughter shapes are visible or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool VisibleRecursive  
  
    { get; set; }
```

EShape.WorldShape

World ancestor.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EWorldShape WorldShape  
    { get; }
```

EShape.ZoomX

Current horizontal zooming factor for drawing operations.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
virtual float ZoomX  
    { get; }
```

EShape.ZoomY

Current vertical zooming factor for drawing operations.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
virtual float ZoomY
{ get; }
```

4.182. ESimpleCropper Class

Manages a point cloud cropper based on X/Y/Z value ranges.

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

XRange

Allowed value range along the X axis. Pass NULL if no cropping should be done along this axis.

YRange

Allowed value range along the Y axis. Pass NULL if no cropping should be done along this axis.

ZRange

M Allowed value range along the Z axis. Pass NULL if no cropping should be done along this axis.

e

Methods

Crop

Crops a point cloud.

ESimpleCropper

Creates an [ESimpleCropper](#) object.

operator=

| Assignment operator

S

impleCropper.Crop

Crops a point cloud.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
void Crop(  
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud cloudIn,  
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud cloudOut  
)
```

Parameters

cloudIn

Cloud to be cropped.

cloudOut

Cropped cloud.

ESimpleCropper.ESimpleCropper

Creates an [ESimpleCropper](#) object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
void ESimpleCropper(  
    )  
  
void ESimpleCropper(  
    Euresys.Open_eVision_2_10.EFloatRange rangeX,  
    Euresys.Open_eVision_2_10.EFloatRange rangeY,  
    Euresys.Open_eVision_2_10.EFloatRange rangeZ  
    )  
  
void ESimpleCropper(  
    Euresys.Open_eVision_2_10.Easy3D.ESimpleCropper other  
    )
```

Parameters

rangeX

Allowed value range along the X axis. Pass NULL if no cropping should be done along this axis.

rangeY

Allowed value range along the Y axis. Pass NULL if no cropping should be done along this axis.

rangeZ

Allowed value range along the Z axis. Pass NULL if no cropping should be done along this axis.

other

An other [ESimpleCropper](#)

ESimpleCropper.operator=

Assignment operator

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.Easy3D.ESimpleCropper operator=(  
    Euresys.Open_eVision_2_10.Easy3D.ESimpleCropper other  
    )
```

Parameters

other

An other [ESimpleCropper](#)

ESimpleCropper.XRange

Allowed value range along the X axis. Pass NULL if no cropping should be done along this axis.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.EFloatRange XRange  
{ get; set; }
```

ESimpleCropper.YRange

Allowed value range along the Y axis. Pass NULL if no cropping should be done along this axis.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.EFloatRange YRange  
{ get; set; }
```

ESimpleCropper.ZRange

Allowed value range along the Z axis. Pass NULL if no cropping should be done along this axis.

Namespace: Euresys.Open_eVision_2_10.Easy3D


```
[C#]
```

```
Euresys.Open_eVision_2_10.EFloatRange ZRange
```

```
{ get; set; }
```

4.183. ESphericalCropper Class

Manages a spherical point cloud cropper.

Namespace: Euresys.Open_eVision_2_10.Easy3D

Methods

Crop

Crops a point cloud.

ESphericalCropper

Creates an [ESphericalCropper](#) object.

operator=

Assignment operator.

S

ESphericalCropper.Crop

Crops a point cloud.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Crop(
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud cloudIn,
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud cloudOut,
    bool invertCrop
)
```

Parameters

cloudIn

Cloud to be cropped.

cloudOut

Cropped cloud.

invertCrop

Indicates if the points kept must be the points inside (true) or outside (false) the sphere.

ESphericalCropper.ESphericalCropper

Creates an [ESphericalCropper](#) object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ESphericalCropper(
    ref Euresys.Open_eVision_2_10.Easy3D.E3DPoint center,
    float radius
)

void ESphericalCropper(
    Euresys.Open_eVision_2_10.Easy3D.ESphericalCropper other
)
```

Parameters

center

Center of the sphere.

radius

Radius of the sphere.

other

An other [ESphericalCropper](#).

ESphericalCropper.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.ESphericalCropper operator=(
    Euresys.Open_eVision_2_10.Easy3D.ESphericalCropper other
)
```

Parameters

other

An other [ESphericalCropper](#).

4.184. EStatistics Class

Calculates various statistics on the pixels values of an [EDepthMap](#) or [EZMap](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

Methods

ComputeAverageMap

Given an input [EDepthMap](#) or [EZMap](#), calculates a depthmap where every pixel is the average of the input depthmap pixels within a window centered on that pixel.

ComputePixelStatistics

Calculates the minimum, maximum, the average and optionally the standard deviation of the pixels values of an [EDepthMap](#) or [EZMap](#).
[EStatistics::ComputePixelStatistics](#) does not take the resolution of the depthmap into account: it calculates statistics on the pixels gray values.

ComputeStandardDeviationMap

Given an input [EDepthMap](#) or [EZMap](#), calculates an image where every pixel is the standard deviation of the input depthmap or zmap pixels within a window centered on that pixel.

ComputeStatistics

Calculates the minimum, maximum, the average and optionally the standard deviation of an [EDepthMap](#) or [EZMap](#) values. The standard deviation is optionally calculated. [EStatistics::ComputeStatistics](#) takes the resolution of the depthmap or the resolution of the Zmap into account: it calculates statistics on the metric values:

Statistics.ComputeAverageMap

Given an input [EDepthMap](#) or [EZMap](#), calculates a depthmap where every pixel is the average of the input depthmap pixels within a window centered on that pixel.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ComputeAverageMap (
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 sourceMap,
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 destinationMap,
    short halfKernelSize,
    float minValidRatio,
    bool fillUndefinedPixels
)
```

```

void ComputeAverageMap (
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 sourceMap,
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 destinationMap,
    short halfKernelSize,
    float minValidRatio,
    bool fillUndefinedPixels
)

void ComputeAverageMap (
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 sourceMap,
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 destinationMap,
    short halfKernelSize,
    float minValidRatio,
    bool fillUndefinedPixels
)

void ComputeAverageMap (
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 sourceMap,
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 destinationMap,
    short halfKernelSize,
    float minValidRatio,
    bool fillUndefinedPixels
)

```

Parameters

sourceMap

The input depthmap/Zmap.

destinationMap

The destination depthmap/Zmap. It should have the same dimensions as the input depthmap.

halfKernelSize

The half-size of the window used for the calculation of the standard deviation. The filter window size (= kernel size) is $\text{halfKernelSize} * 2 + 1$, should be positive, smaller than (or equal to) the image size, and may not exceed 256.

minValidRatio

required ratio of valid pixels in the filter window to process the calculation. If not enough, the output pixel will be marked as invalid. The default value of this parameter is 0.25

fillUndefinedPixels

This boolean controls how undefined pixels are handled: either they filled by the calculated average value, or they are left undefined (default behavior).

EStatistics.ComputePixelStatistics

Calculates the minimum, maximum, the average and optionally the standard deviation of the pixels values of an [EDepthMap](#) or [EZMap](#).

[EStatistics::ComputePixelStatistics](#) does not take the resolution of the depthmap into account: it calculates statistics on the pixels gray values.

Namespace: Euresys.Open_eVision_2_10.Easy3D

[C#]

```
void ComputePixelStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 sourceMap,  
    ref uint validCount,  
    ref Euresys.Open_eVision_2_10.EBW8 minimumValue,  
    ref Euresys.Open_eVision_2_10.EBW8 maximumValue,  
    ref float average  
)  
  
void ComputePixelStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 sourceMap,  
    ref uint validCount,  
    ref Euresys.Open_eVision_2_10.EBW16 minimumValue,  
    ref Euresys.Open_eVision_2_10.EBW16 maximumValue,  
    ref float average  
)  
  
void ComputePixelStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap32f sourceMap,  
    ref uint validCount,  
    ref Euresys.Open_eVision_2_10.EBW32f minimumValue,  
    ref Euresys.Open_eVision_2_10.EBW32f maximumValue,  
    ref float average  
)
```

```
void ComputePixelStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 sourceMap,  
    Euresys.Open_eVision_2_10.ERegion region,  
    ref uint validCount,  
    ref Euresys.Open_eVision_2_10.EBW16 minimumValue,  
    ref Euresys.Open_eVision_2_10.EBW16 maximumValue,  
    ref float average  
)
```

```
void ComputePixelStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 sourceMap,  
    Euresys.Open_eVision_2_10.ERegion region,  
    ref uint validCount,  
    ref Euresys.Open_eVision_2_10.EBW8 minimumValue,  
    ref Euresys.Open_eVision_2_10.EBW8 maximumValue,  
    ref float average  
)
```

```
void ComputePixelStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap32f sourceMap,  
    Euresys.Open_eVision_2_10.ERegion region,  
    ref uint validCount,  
    ref Euresys.Open_eVision_2_10.EBW32f minimumValue,  
    ref Euresys.Open_eVision_2_10.EBW32f maximumValue,  
    ref float average  
)
```

```
void ComputePixelStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 sourceMap,  
    ref uint validCount,  
    ref Euresys.Open_eVision_2_10.EBW8 minimumValue,  
    ref Euresys.Open_eVision_2_10.EBW8 maximumValue,  
    ref float average,  
    ref float stddev  
)
```

```
void ComputePixelStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 sourceMap,  
    Euresys.Open_eVision_2_10.ERegion region,  
    ref uint validCount,  
    ref Euresys.Open_eVision_2_10.EBW8 minimumValue,  
    ref Euresys.Open_eVision_2_10.EBW8 maximumValue,  
    ref float average,  
    ref float stddev  
)
```

```
void ComputePixelStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 sourceMap,  
    ref uint validCount,  
    ref Euresys.Open_eVision_2_10.EBW16 minimumValue,  
    ref Euresys.Open_eVision_2_10.EBW16 maximumValue,  
    ref float average,  
    ref float stddev  
)
```

```
void ComputePixelStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 sourceMap,  
    Euresys.Open_eVision_2_10.ERegion region,  
    ref uint validCount,  
    ref Euresys.Open_eVision_2_10.EBW16 minimumValue,  
    ref Euresys.Open_eVision_2_10.EBW16 maximumValue,  
    ref float average,  
    ref float stddev  
)
```

```
void ComputePixelStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap32f sourceMap,  
    ref uint validCount,  
    ref Euresys.Open_eVision_2_10.EBW32f minimumValue,  
    ref Euresys.Open_eVision_2_10.EBW32f maximumValue,  
    ref float average,  
    ref float stddev  
)
```

```
void ComputePixelStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap32f sourceMap,  
    Euresys.Open_eVision_2_10.ERegion region,  
    ref uint validCount,  
    ref Euresys.Open_eVision_2_10.EBW32f minimumValue,  
    ref Euresys.Open_eVision_2_10.EBW32f maximumValue,  
    ref float average,  
    ref float stddev  
)
```

```
void ComputePixelStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 sourceMap,  
    ref uint validCount,  
    ref Euresys.Open_eVision_2_10.EBW8 minimumValue,  
    ref Euresys.Open_eVision_2_10.EBW8 maximumValue,  
    ref float average  
)
```



```
void ComputePixelStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 sourceMap,  
    Euresys.Open_eVision_2_10.ERegion region,  
    ref uint validCount,  
    ref Euresys.Open_eVision_2_10.EBW8 minimumValue,  
    ref Euresys.Open_eVision_2_10.EBW8 maximumValue,  
    ref float average  
)  
  
void ComputePixelStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 sourceMap,  
    ref uint validCount,  
    ref Euresys.Open_eVision_2_10.EBW16 minimumValue,  
    ref Euresys.Open_eVision_2_10.EBW16 maximumValue,  
    ref float average  
)  
  
void ComputePixelStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 sourceMap,  
    Euresys.Open_eVision_2_10.ERegion region,  
    ref uint validCount,  
    ref Euresys.Open_eVision_2_10.EBW16 minimumValue,  
    ref Euresys.Open_eVision_2_10.EBW16 maximumValue,  
    ref float average  
)  
  
void ComputePixelStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap32f sourceMap,  
    ref uint validCount,  
    ref Euresys.Open_eVision_2_10.EBW32f minimumValue,  
    ref Euresys.Open_eVision_2_10.EBW32f maximumValue,  
    ref float average  
)  
  
void ComputePixelStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap32f sourceMap,  
    Euresys.Open_eVision_2_10.ERegion region,  
    ref uint validCount,  
    ref Euresys.Open_eVision_2_10.EBW32f minimumValue,  
    ref Euresys.Open_eVision_2_10.EBW32f maximumValue,  
    ref float average  
)
```

```
void ComputePixelStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 sourceMap,  
    ref uint validCount,  
    ref Euresys.Open_eVision_2_10.EBW8 minimumValue,  
    ref Euresys.Open_eVision_2_10.EBW8 maximumValue,  
    ref float average,  
    ref float stddev  
)  
  
void ComputePixelStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 sourceMap,  
    Euresys.Open_eVision_2_10.ERegion region,  
    ref uint validCount,  
    ref Euresys.Open_eVision_2_10.EBW8 minimumValue,  
    ref Euresys.Open_eVision_2_10.EBW8 maximumValue,  
    ref float average,  
    ref float stddev  
)  
  
void ComputePixelStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 sourceMap,  
    ref uint validCount,  
    ref Euresys.Open_eVision_2_10.EBW16 minimumValue,  
    ref Euresys.Open_eVision_2_10.EBW16 maximumValue,  
    ref float average,  
    ref float stddev  
)  
  
void ComputePixelStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 sourceMap,  
    Euresys.Open_eVision_2_10.ERegion region,  
    ref uint validCount,  
    ref Euresys.Open_eVision_2_10.EBW16 minimumValue,  
    ref Euresys.Open_eVision_2_10.EBW16 maximumValue,  
    ref float average,  
    ref float stddev  
)
```

```

void ComputePixelStatistics(
    Euresys.Open_eVision_2_10.Easy3D.EZMap32f sourceMap,
    ref uint validCount,
    ref Euresys.Open_eVision_2_10.EBW32f minimumValue,
    ref Euresys.Open_eVision_2_10.EBW32f maximumValue,
    ref float average,
    ref float stddev
)

void ComputePixelStatistics(
    Euresys.Open_eVision_2_10.Easy3D.EZMap32f sourceMap,
    Euresys.Open_eVision_2_10.ERegion region,
    ref uint validCount,
    ref Euresys.Open_eVision_2_10.EBW32f minimumValue,
    ref Euresys.Open_eVision_2_10.EBW32f maximumValue,
    ref float average,
    ref float stddev
)

```

Parameters

sourceMap

The input depthmap/Zmap.

validCount

Variable to store the number of valid pixels in sourceMap.

minimumValue

Variable to store the minimum value.

maximumValue

Variable to store the maximum value.

average

Variable to store the average value.

region

The [ERegion](#) where the statistics has to be calculated.

stddev

Variable to store the standard deviation.

EStatistics.ComputeStandardDeviationMap

Given an input [EDepthMap](#) or [EZMap](#), calculates an image where every pixel is the standard deviation of the input depthmap or zmap pixels within a window centered on that pixel.

Namespace: Euresys.Open_eVision_2_10.Easy3D

[C#]

```
void ComputeStandardDeviationMap(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 sourceMap,  
    Euresys.Open_eVision_2_10.EImageBW8 destinationImage,  
    short halfKernelSize,  
    float minValidRatio,  
    bool fillUndefinedPixels  
)  
  
void ComputeStandardDeviationMap(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 sourceMap,  
    Euresys.Open_eVision_2_10.EImageBW16 destinationImage,  
    short halfKernelSize,  
    float minValidRatio,  
    bool fillUndefinedPixels  
)  
  
void ComputeStandardDeviationMap(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 sourceMap,  
    Euresys.Open_eVision_2_10.EImageBW16 destinationImage,  
    short halfKernelSize,  
    float minValidRatio,  
    bool fillUndefinedPixels  
)  
  
void ComputeStandardDeviationMap(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 sourceMap,  
    Euresys.Open_eVision_2_10.EImageBW16 destinationImage,  
    short halfKernelSize,  
    float minValidRatio,  
    bool fillUndefinedPixels  
)
```

```
void ComputeStandardDeviationMap(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 sourceMap,  
    Euresys.Open_eVision_2_10.EImageBW8 destinationImage,  
    short halfKernelSize,  
    float minValidRatio,  
    bool fillUndefinedPixels  
)  
  
void ComputeStandardDeviationMap(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 sourceMap,  
    Euresys.Open_eVision_2_10.EImageBW16 destinationImage,  
    short halfKernelSize,  
    float minValidRatio,  
    bool fillUndefinedPixels  
)
```

Parameters

sourceMap

The input depthmap/Zmap.

destinationImage

The destination image. It should have the same dimensions as the input depthmap.

halfKernelSize

The half-size of the window used for the calculation of the standard deviation.

The filter window size (= kernel size) is $\text{halfKernelSize} * 2 + 1$, should be positive, smaller than (or equal to) the image size, and may not exceed 256.

minValidRatio

required ratio of valid pixels in the filter window to process the calculation.

If not enough, the output pixel value will be 0. The default value of this parameter is 0.25 .

fillUndefinedPixels

This boolean controls how undefined pixels are handled: either they filled by the calculated standard deviation, or they are left undefined (default behavior).

Remarks

When the input depthmap is on 8 bits and the destination image is on 16 bits, the resulting standard deviation scale is 256 times larger than in the source depthmap.

EStatistics.ComputeStatistics

Calculates the minimum, maximum, the average and optionally the standard deviation of an [EDepthMap](#) or [EZMap](#) values.

The standard deviation is optionally calculated. [EStatistics::ComputeStatistics](#) takes the resolution of the depthmap or the resolution of the Zmap into account: it calculates statistics on the metric values:

Namespace: Euresys.Open_eVision_2_10.Easy3D

[C#]

```
void ComputeStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 sourceMap,  
    ref uint validCount,  
    ref float minimumValue,  
    ref float maximumValue,  
    ref float average  
)  
  
void ComputeStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 sourceMap,  
    Euresys.Open_eVision_2_10.ERegion region,  
    ref uint validCount,  
    ref float minimumValue,  
    ref float maximumValue,  
    ref float average  
)  
  
void ComputeStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 sourceMap,  
    ref uint validCount,  
    ref float minimumValue,  
    ref float maximumValue,  
    ref float average  
)
```

```
void ComputeStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 sourceMap,  
    Euresys.Open_eVision_2_10.ERegion region,  
    ref uint validCount,  
    ref float minimumValue,  
    ref float maximumValue,  
    ref float average  
)  
  
void ComputeStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap32f sourceMap,  
    ref uint validCount,  
    ref float minimumValue,  
    ref float maximumValue,  
    ref float average  
)  
  
void ComputeStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap32f sourceMap,  
    Euresys.Open_eVision_2_10.ERegion region,  
    ref uint validCount,  
    ref float minimumValue,  
    ref float maximumValue,  
    ref float average  
)  
  
void ComputeStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 sourceMap,  
    ref uint validCount,  
    ref float minimumValue,  
    ref float maximumValue,  
    ref float average  
)  
  
void ComputeStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 sourceMap,  
    Euresys.Open_eVision_2_10.ERegion region,  
    ref uint validCount,  
    ref float minimumValue,  
    ref float maximumValue,  
    ref float average  
)
```

```
void ComputeStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 sourceMap,  
    ref uint validCount,  
    ref float minimumValue,  
    ref float maximumValue,  
    ref float average  
)  
  
void ComputeStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 sourceMap,  
    Euresys.Open_eVision_2_10.ERegion region,  
    ref uint validCount,  
    ref float minimumValue,  
    ref float maximumValue,  
    ref float average  
)  
  
void ComputeStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap32f sourceMap,  
    ref uint validCount,  
    ref float minimumValue,  
    ref float maximumValue,  
    ref float average  
)  
  
void ComputeStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap32f sourceMap,  
    Euresys.Open_eVision_2_10.ERegion region,  
    ref uint validCount,  
    ref float minimumValue,  
    ref float maximumValue,  
    ref float average  
)  
  
void ComputeStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 sourceMap,  
    ref uint validCount,  
    ref float minimumValue,  
    ref float maximumValue,  
    ref float average,  
    ref float stddev  
)
```



```
void ComputeStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap8 sourceMap,  
    Euresys.Open_eVision_2_10.ERegion region,  
    ref uint validCount,  
    ref float minimumValue,  
    ref float maximumValue,  
    ref float average,  
    ref float stddev  
)  
  
void ComputeStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 sourceMap,  
    ref uint validCount,  
    ref float minimumValue,  
    ref float maximumValue,  
    ref float average,  
    ref float stddev  
)  
  
void ComputeStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap16 sourceMap,  
    Euresys.Open_eVision_2_10.ERegion region,  
    ref uint validCount,  
    ref float minimumValue,  
    ref float maximumValue,  
    ref float average,  
    ref float stddev  
)  
  
void ComputeStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap32f sourceMap,  
    ref uint validCount,  
    ref float minimumValue,  
    ref float maximumValue,  
    ref float average,  
    ref float stddev  
)
```

```
void ComputeStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EDepthMap32f sourceMap,  
    Euresys.Open_eVision_2_10.ERegion region,  
    ref uint validCount,  
    ref float minimumValue,  
    ref float maximumValue,  
    ref float average,  
    ref float stddev  
)
```

```
void ComputeStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 sourceMap,  
    ref uint validCount,  
    ref float minimumValue,  
    ref float maximumValue,  
    ref float average,  
    ref float stddev  
)
```

```
void ComputeStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 sourceMap,  
    Euresys.Open_eVision_2_10.ERegion region,  
    ref uint validCount,  
    ref float minimumValue,  
    ref float maximumValue,  
    ref float average,  
    ref float stddev  
)
```

```
void ComputeStatistics(  
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 sourceMap,  
    ref uint validCount,  
    ref float minimumValue,  
    ref float maximumValue,  
    ref float average,  
    ref float stddev  
)
```

```

void ComputeStatistics(
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 sourceMap,
    Euresys.Open_eVision_2_10.ERegion region,
    ref uint validCount,
    ref float minimumValue,
    ref float maximumValue,
    ref float average,
    ref float stddev
)

void ComputeStatistics(
    Euresys.Open_eVision_2_10.Easy3D.EZMap32f sourceMap,
    ref uint validCount,
    ref float minimumValue,
    ref float maximumValue,
    ref float average,
    ref float stddev
)

void ComputeStatistics(
    Euresys.Open_eVision_2_10.Easy3D.EZMap32f sourceMap,
    Euresys.Open_eVision_2_10.ERegion region,
    ref uint validCount,
    ref float minimumValue,
    ref float maximumValue,
    ref float average,
    ref float stddev
)

```

Parameters

sourceMap

The input depthmap/Zmap.

validCount

Variable to store the number of valid pixels in sourceMap.

minimumValue

Variable to store the minimum value.

maximumValue

Variable to store the maximum value.

average

Variable to store the average value.

region

The [ERegion](#) where the statistics has to be calculated.

stddev

Variable to store the standard deviation value.

4.185. EStringPair Class

Represent a Key/Value pair of strings.

Namespace: Euresys.Open_eVision_2_10

Properties

Key

Returns the key.

Value

M Returns the value.

e

Methods

EStringPair

Constructs an EStringPair context.

operator=

E Assignment operator

S

tringPair.EStringPair

Constructs an EStringPair context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EStringPair(
    string key,
    string value
)

void EStringPair(
    Euresys.Open_eVision_2_10.EStringPair other
)

void EStringPair(
)
```

Parameters

key

The key.

value

The value associated to the key.

other

-

EStringPair.Key

Returns the key.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
string Key
{ get; }
```

EStringPair.operator=

Assignment operator

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EStringPair operator=(  
    Euresys.Open_eVision_2_10.EStringPair other  
)
```

Parameters

other

-

EStringPair.Value

Returns the value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
string Value  
    { get; }
```

4.186. EThreeLayersImageSegmenter Class

The base class from which all the segmenters that produce three layers derive.

Remarks

The Layer Encoding can be enabled/disabled for each layer individually. The index of the layers in the coded image can also be assigned individually.

Base Class: [EImageSegmenter](#)

Derived Class(es): [EGrayscaleDoubleThresholdSegmenter](#)

Namespace: Euresys.Open_eVision_2_10.Segmenters

Properties

[BlackLayerEncoded](#)

Black layer encoding status.

[BlackLayerIndex](#)

Index of the black layer in the destination coded image.

[NeutralLayerEncoded](#)

Neutral layer encoding status.

[NeutralLayerIndex](#)

Index of the neutral layer in the destination coded image.

[WhiteLayerEncoded](#)

White layer encoding status.

[WhiteLayerIndex](#)

Index of the white layer in the destination coded image.

T

hreeLayersImageSegmenter.BlackLayerEncoded

Black layer encoding status.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]
virtual bool BlackLayerEncoded
    { get; set; }
```

EThreeLayersImageSegmenter.BlackLayerIndex

Index of the black layer in the destination coded image.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]
virtual uint BlackLayerIndex
    { get; set; }
```

Remarks

Setting this property automatically switches on the encoding of the black layer.

EThreeLayersImageSegmenter.NeutralLayerEncoded

Neutral layer encoding status.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]
virtual bool NeutralLayerEncoded
    { get; set; }
```


EThreeLayersImageSegmenter.NeutralLayerIndex

Index of the neutral layer in the destination coded image.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]  
virtual uint NeutralLayerIndex  
  
    { get; set; }
```

Remarks

Setting this property automatically switches on the encoding of the neutral layer.

EThreeLayersImageSegmenter.WhiteLayerEncoded

White layer encoding status.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]  
virtual bool WhiteLayerEncoded  
  
    { get; set; }
```

EThreeLayersImageSegmenter.WhiteLayerIndex

Index of the white layer in the destination coded image.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]
virtual uint WhiteLayerIndex
{ get; set; }
```

Remarks

Setting this property automatically switches on the encoding of the white layer.

4.187. ETwoLayersImageSegmenter Class

The base class from which all the segmenters that produce two layers derive.

Remarks

The Layer Encoding can be enabled/disabled for each layer individually. The index of the layers in the coded image can also be assigned individually.

Base Class: [EImageSegmenter](#)

Derived Class(es): [EBinaryImageSegmenter](#) [EColorRangeThresholdSegmenter](#) [EColorSingleThresholdSegmenter](#) [EGrayscaleSingleThresholdSegmenter](#) [EImageRangeSegmenter](#) [EReferenceImageSegmenter](#)

Namespace: Euresys.Open_eVision_2_10.Segmenters

Properties

[BlackLayerEncoded](#)

Black layer encoding status.

[BlackLayerIndex](#)

Index of the black layer in the destination coded image.

[WhiteLayerEncoded](#)

White layer encoding status.

[WhiteLayerIndex](#)

Index of the white layer in the destination coded image.

ETwoLayersImageSegmenter.BlackLayerEncoded

Black layer encoding status.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]  
virtual bool BlackLayerEncoded  
    { get; set; }
```

ETwoLayersImageSegmenter.BlackLayerIndex

Index of the black layer in the destination coded image.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]  
virtual uint BlackLayerIndex  
    { get; set; }
```

Remarks

Setting this property automatically switches on the encoding of the black layer.

ETwoLayersImageSegmenter.WhiteLayerEncoded

White layer encoding status.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]
virtual bool WhiteLayerEncoded
{ get; set; }
```

ETwoLayersImageSegmenter.WhiteLayerIndex

Index of the white layer in the destination coded image.

Namespace: Euresys.Open_eVision_2_10.Segmenters

```
[C#]
virtual uint WhiteLayerIndex
{ get; set; }
```

Remarks

Setting this property automatically switches on the encoding of the white layer.

4.188. EUnwarpingLut Class

This class is used only as a lookup table in the [EWorldShape::Unwarp](#) and [EWorldShape::SetupUnwarp](#) methods. It has no other use of its own.

Namespace: Euresys.Open_eVision_2_10

Methods

EUnwarpingLut

Constructs a EUnwarpingLut object that is used to speed-up the unwarping process.

EUnwarpingLut.EUnwarpingLut

Constructs a EUnwarpingLut object that is used to speed-up the unwarping process.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EUnwarpingLut(
    Euresys.Open_eVision_2_10.EUnwarpingLut other
)
void EUnwarpingLut(
)
```

Parameters

other

-

4.189. EVector Class

Base class for all typed vectors.

Remarks

This class contains all methods that are not type specific. Mainly methods to handle elements count and serialization

Derived Class(es): [EBW32Vector](#) [EPathVector](#) [EBW16PathVector](#) [EBW16Vector](#) [EBW8PathVector](#) [EBW8Vector](#) [EBWHistogramVector](#) [EC24PathVector](#) [EC24Vector](#) [EColorVector](#) [EPeakVector](#)

Namespace: Euresys.Open_eVision_2_10

Properties

[NumElements](#)

Number of elements in the vector.

Methods

Empty

Resets the number of elements to **0**.

RemoveElement

Removes the element at the specified position

Serialize

E-

V

ector.Empty

Resets the number of elements to **0**.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Empty(  
)
```

EVector.NumElements

Number of elements in the vector.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint NumElements
{ get; set; }
```

EVector.RemoveElement

Removes the element at the specified position

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void RemoveElement(
    uint index
)
```

Parameters

index

Index, between **0** and [EVector::NumElements](#) (excluded) of the element to be accessed.

EVector.Serialize

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void Serialize(  
    Euresys.Open_eVision_2_10.ESerializer serializer,  
    uint un32Version  
)
```

Parameters

serializer

-

un32Version

-

4.190. EWedge Class

Represents a model of a wedge (disk, ring, sector or curvilinear quadrilateral) in EasyGauge.

Base Class: [EFrame](#)

Namespace: Euresys.Open_eVision_2_10

Properties

[Amplitude](#)

Angular amplitude of the [EWedge](#).

[ApexAngle](#)

Angular position at the apex of the [EWedge](#).

[Breadth](#)

Breadth of the [EWedge](#).

[Direct](#)

Flag indicating whether positive angles correspond to a clockwise or an anticlockwise rotation.

EndAngle	Ending angular position of the EWedge.
FullBreadth	Flag indicating if the EWedge has a full breadth (breadth = radius).
FullCircle	Flag indicating if the EWedge is a full circle (amplitude = 2 PI).
InnerApex	Inner apex point coordinates of the EWedge.
InnerArcLength	Inner circle arc length of the EWedge.
InnerDiameter	Inner diameter of the EWedge.
InnerEnd	Inner end point coordinates of the EWedge.
InnerOrg	Inner origin point coordinates of the EWedge.
InnerRadius	Inner radius of the EWedge
OrgAngle	Angular position from where the EWedge extents.
OuterApex	Outer apex point coordinates of the EWedge.
OuterArcLength	Outer circle arc length of the EWedge.

OuterDiameter

Outer diameter of the [EWedge](#).

OuterEnd

Outer end point coordinates of the [EWedge](#).

OuterOrg

Outer origin point coordinates of the [EWedge](#).

OuterRadius

M Outer radius of the [EWedge](#).

e

thods

CopyTo

Copies all the data of the current [EWedge](#) object into another [EWedge](#) object and returns it.

EWedge

Constructs a [EWedge](#) object.

GetCorners

Retrieves the coordinates of each corner of the [EWedge](#).

GetEdges

Retrieves each edge of the [EWedge](#).

GetInnerPoint

Returns the coordinates of a particular point specified by its location along the inner circle arc.

GetMidEdges

Retrieves the center coordinates of each edge of the wedge fitting gauge.

[GetOuterPoint](#)

Returns the coordinates of a particular point specified by its location along the outer circle arc.

[GetPoint](#)

Returns the coordinates of a particular point specified by its location along the wedge perimeter.

[operator=](#)

Copies all the data from another [EWedge](#) object into the current [EWedge](#) object

[SetDiameters](#)

Sets the nominal inner/outer diameter and breadth of the [EWedge](#).

[SetFromCenterAndOrigin](#)

Sets the geometric parameters (nominal position and diameter, breadth, angular position and amplitude) of an [EWedge](#) object.

[SetFromOriginMiddleEnd](#)

Sets the geometric parameters (nominal position and diameter, breadth, angular position and amplitude) of an [EWedge](#) object.

[SetFromTwoPoints](#)

DEPRECATED (you should use [EWedge::SetFromCenterAndOrigin](#)): Sets the geometric parameters (nominal position and diameter, breadth, angular position and amplitude) of an [EWedge](#) object.

[SetRadii](#)

Sets the nominal radius and breadth of the [EWedge](#).

EWedge.Amplitude

Angular amplitude of the [EWedge](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Amplitude  
  
{ get; set; }
```

Remarks

A [EWedge](#) is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), its breadth, the angular position from where it extends, its angular amplitude, and its outline tolerance.

The sign of the rotation angle depends whether the field of view is calibrated or not.

* When the field of view is calibrated, the coordinate system is said to be direct, the abscissa extends rightwards and the ordinate extends upwards. In this case, an anticlockwise rotation leads to a positive angle value.

* When the field of view is not calibrated, the coordinate system is said to be inverse, the abscissa extends rightwards and the ordinate extends downwards. In this case, a clockwise rotation leads to a positive angle value.

EWedge.ApexAngle

Angular position at the apex of the [EWedge](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float ApexAngle  
  
{ get; }
```

Remarks

A [EWedge](#) is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), its breadth, the angular position from where it extents, its angular amplitude, and its outline tolerance.

The sign of the rotation angle depends whether the field of view is calibrated or not.

* When the field of view is calibrated, the coordinate system is said to be direct, the abscissa extends rightwards and the ordinate extends upwards. In this case, an anticlockwise rotation leads to a positive angle value.

* When the field of view is not calibrated, the coordinate system is said to be inverse, the abscissa extends rightwards and the ordinate extends downwards. In this case, a clockwise rotation leads to a positive angle value.

EWedge.Breadth

Breadth of the [EWedge](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
float Breadth  
  
{ get; }
```

Remarks

A [EWedge](#) is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), its breadth, the angular position from where it extents, its angular amplitude, and its outline tolerance.

EWedge.CopyTo

Copies all the data of the current [EWedge](#) object into another [EWedge](#) object and returns it.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EWedge CopyTo(  
    Euresys.Open_eVision_2_10.EWedge destinationImage  
)
```

Parameters

destinationImage

Pointer to the [EWedge](#) object in which the current [EWedge](#) object data have to be copied.

Remarks

In case of a **NULL** pointer, a new [EWedge](#) object will be created and returned.

EWedge.Direct

Flag indicating whether positive angles correspond to a clockwise or an anticlockwise rotation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
bool Direct  
  
    { get; }
```

Remarks

TRUE (default) means that angles increase anticlockwisely in a direct coordinate system, and clockwisely in an inverse coordinate system.

* When the field of view is calibrated, the coordinate system is said to be direct, the abscissa extends rightwards and the ordinate extends upwards.

* When the field of view is not calibrated, the coordinate system is said to be inverse the abscissa extends rightwards and the ordinate extends downwards.

EWedge.EndAngle

Ending angular position of the [EWedge](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
float EndAngle  
  
    { get; }
```

Remarks

A [EWedge](#) is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), its breadth, the angular position from where it extends, its angular amplitude, and its outline tolerance.

The sign of the rotation angle depends whether the field of view is calibrated or not.

* When the field of view is calibrated, the coordinate system is said to be direct, the abscissa extends rightwards and the ordinate extends upwards. In this case, an anticlockwise rotation leads to a positive angle value.

* When the field of view is not calibrated, the coordinate system is said to be inverse, the abscissa extends rightwards and the ordinate extends downwards. In this case, a clockwise rotation leads to a positive angle value.

EWedge.EWedge

Constructs a [EWedge](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void EWedge (  
    )  
  
void EWedge (  
    Euresys.Open_eVision_2_10.EPoint center,  
    float diameter,  
    float breadth,  
    float originAngle,  
    bool direct  
    )
```

```

void EWedge (
    Euresys.Open_eVision_2_10.EPoint center,
    Euresys.Open_eVision_2_10.EPoint origin,
    float breadth,
    bool direct
)

void EWedge (
    Euresys.Open_eVision_2_10.EPoint center,
    float diameter,
    float breadth,
    float originAngle,
    float amplitude
)

void EWedge (
    Euresys.Open_eVision_2_10.EPoint origin,
    Euresys.Open_eVision_2_10.EPoint middle,
    Euresys.Open_eVision_2_10.EPoint end,
    float breadth,
    bool fullCircle
)

void EWedge (
    Euresys.Open_eVision_2_10.EWedge other
)

```

Parameters

center

Center coordinates of the wedge at its nominal position. The default value is **(0,0)**.

diameter

Nominal diameter of the wedge. The default value is **100**.

breadth

Nominal breadth of the wedge. It must be negative or zero. The default value is **-50**.

originAngle

Origin point coordinates of the wedge.

direct

TRUE (default) means that angles increase anticlockwise in a direct coordinate system and clockwise in an inverse coordinate system.

origin

Origin point coordinates of the wedge.

amplitude

Nominal angular amplitude of the wedge. The default value is **360**.

middle

Middle point coordinates of the wedge.

end

End point coordinates of the wedge.

fullCircle

TRUE (default) in case of a full turn wedge. If **fullCircle** is **FALSE**, **origin** and **end** give the wedge's amplitude.

other

Another [EWedge](#) object to be copied in the new [EWedge](#) object.

EWedge.FullBreadth

Flag indicating if the [EWedge](#) has a full breadth (**breadth = radius**).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool FullBreadth  
  
{ get; }
```

EWedge.FullCircle

Flag indicating if the [EWedge](#) is a full circle (**amplitude = 2 PI**).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool FullCircle  
  
{ get; }
```

EWedge.GetCorners

Retrieves the coordinates of each corner of the [EWedge](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetCorners (
    Euresys.Open_eVision_2_10.EPoint ar,
    Euresys.Open_eVision_2_10.EPoint AAr,
    Euresys.Open_eVision_2_10.EPoint aRR,
    Euresys.Open_eVision_2_10.EPoint AARR
)
```

Parameters

ar

Coordinates of the inner org corner of the [EWedge](#).

AAr

Coordinates of the inner end corner of the [EWedge](#).

aRR

Coordinates of the outer org corner of the [EWedge](#).

AARR

Coordinates of the outer end corner of the [EWedge](#).

EWedge.GetEdges

Retrieves each edge of the [EWedge](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetEdges (
    Euresys.Open_eVision_2_10.ELine a,
    Euresys.Open_eVision_2_10.ELine AA,
    Euresys.Open_eVision_2_10.ECircle r,
    Euresys.Open_eVision_2_10.ECircle RR
)
```

Parameters

a

Org edge of the [EWedge](#).

AA

End edge of the [EWedge](#).

r

Inner edge of the [EWedge](#).

RR

Outer edge of the [EWedge](#).

EWedge.GetInnerPoint

Returns the coordinates of a particular point specified by its location along the inner circle arc.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint GetInnerPoint(
    float fraction
)
```

Parameters

fraction

Point location expressed as a fraction of the circle arc (range **[-1, +1]**).

EWedge.GetMidEdges

Retrieves the center coordinates of each edge of the wedge fitting gauge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetMidEdges (
    Euresys.Open_eVision_2_10.EPoint a,
    Euresys.Open_eVision_2_10.EPoint AA,
    Euresys.Open_eVision_2_10.EPoint r,
    Euresys.Open_eVision_2_10.EPoint RR
)
```

Parameters

a

Center coordinates of the org edge of the [EWedge](#).

AA

Center coordinates of the end edge of the [EWedge](#).

r

Center coordinates of the inner edge of the [EWedge](#).

RR

Center coordinates of the outer edge of the [EWedge](#).

EWedge.GetOuterPoint

Returns the coordinates of a particular point specified by its location along the outter circle arc.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint GetOuterPoint(
    float fraction
)
```

Parameters

fraction

Point location expressed as a fraction of the circle arc (range **[-1, +1]**).

EWedge.GetPoint

Returns the coordinates of a particular point specified by its location along the wedge perimeter.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint GetPoint(
    float breadthFraction,
    float angleFraction
)
```

Parameters

breadthFraction

Point location expressed as a fraction of the wedge breadth (range -1, +1).

angleFraction

Point location expressed as a fraction of the wedge amplitude (range -1, +1).

EWedge.InnerApex

Inner apex point coordinates of the [EWedge](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EPoint InnerApex  
  
{ get; }
```

EWedge.InnerArcLength

Inner circle arc length of the [EWedge](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float InnerArcLength  
  
{ get; }
```

EWedge.InnerDiameter

Inner diameter of the [EWedge](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float InnerDiameter  
  
{ get; }
```

EWedge.InnerEnd

Inner end point coordinates of the [EWedge](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint InnerEnd  
    { get; }
```

EWedge.InnerOrg

Inner origin point coordinates of the [EWedge](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint InnerOrg  
    { get; }
```

EWedge.InnerRadius

Inner radius of the [EWedge](#)

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float InnerRadius
{ get; }
```

EWedge.operator=

Copies all the data from another [EWedge](#) object into the current [EWedge](#) object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EWedge operator=(
    Euresys.Open_eVision_2_10.EWedge other
)
```

Parameters

other
[EWedge](#) object to be copied

EWedge.OrgAngle

Angular position from where the [EWedge](#) extends.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```



```
float OrgAngle
```

```
{ get; }
```

Remarks

A [EWedge](#) is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), its breadth, the angular position from where it extents, its angular amplitude, and its outline tolerance.

The sign of the rotation angle depends whether the field of view is calibrated or not.

* When the field of view is calibrated, the coordinate system is said to be direct, the abscissa extends rightwards and the ordinate extends upwards. In this case, an anticlockwise rotation leads to a positive angle value.

* When the field of view is not calibrated, the coordinate system is said to be inverse, the abscissa extends rightwards and the ordinate extends downwards. In this case, a clockwise rotation leads to a positive angle value.

EWedge.OuterApex

Outer apex point coordinates of the [EWedge](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EPoint OuterApex
```

```
{ get; }
```

Remarks

A [EWedge](#) is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), its breadth, the angular position from where it extents, its angular amplitude, and its outline tolerance.

EWedge.OuterArcLength

Outer circle arc length of the [EWedge](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float OuterArcLength
{ get; }
```

Remarks

A [EWedge](#) is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), its breadth, the angular position from where it extents, its angular amplitude, and its outline tolerance.

EWedge.OuterDiameter

Outer diameter of the [EWedge](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float OuterDiameter
{ get; }
```

Remarks

A [EWedge](#) is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), its breadth, the angular position from where it extents, its angular amplitude, and its outline tolerance.

EWedge.OuterEnd

Outer end point coordinates of the [EWedge](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint OuterEnd
{ get; }
```

Remarks

A [EWedge](#) is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), its breadth, the angular position from where it extents, its angular amplitude, and its outline tolerance.

EWedge.OuterOrg

Outer origin point coordinates of the [EWedge](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint OuterOrg
{ get; }
```

Remarks

A [EWedge](#) is fully defined knowing its nominal position (its center coordinates), its nominal outer radius (diameter), its breadth (must be negative), the angular position from where it extents, its angular amplitude, and its outline tolerance.

EWedge.OuterRadius

Outer radius of the [EWedge](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float OuterRadius
{ get; }
```

Remarks

A [EWedge](#) is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), its breadth, the angular position from where it extents, its angular amplitude, and its outline tolerance.

EWedge.SetDiameters

Sets the nominal inner/outer diameter and breadth of the [EWedge](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetDiameters(
    float diameter,
    float breadth
)
```

Parameters

diameter

Outer diameter of the [EWedge](#). The default value is **100**.

breadth

Breadth of the [EWedge](#). It must be negative or zero. Its default value is **-50**.

Remarks

A [EWedge](#) is fully defined knowing its nominal position (given by the coordinates of its center), its nominal outer radius (diameter), its breadth, the angular position from where it extents, its angular amplitude and its outline tolerance.

By default, the [EWedge](#) diameter value is **100**, which means 100 pixels when the field of view is not calibrated and 100 "units" in case of a calibrated field of view.

EWedge.SetFromCenterAndOrigin

Sets the geometric parameters (nominal position and diameter, breadth, angular position and amplitude) of an [EWedge](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetFromCenterAndOrigin(
    Euresys.Open_eVision_2_10.EPoint center,
    Euresys.Open_eVision_2_10.EPoint origin,
    float breadth,
    bool direct
)
```

Parameters

center

Center coordinates of the wedge at its nominal position. The default value is **(0,0)**.

origin

Origin point coordinates of the wedge.

breadth

Nominal breadth of the wedge. It must be negative or zero. The default value is **-50**.

direct

TRUE (default) means that angles increase anticlockwise in a direct coordinate system and clockwise in an inverse coordinate system.

Remarks

In a direct coordinate system, the abscissa extends rightwards and the ordinate extends upwards. The coordinate system is said to be inverse if the abscissa extends rightwards and the ordinate extends downwards.

EWedge.SetFromOriginMiddleEnd

Sets the geometric parameters (nominal position and diameter, breadth, angular position and amplitude) of an [EWedge](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetFromOriginMiddleEnd(
    Euresys.Open_eVision_2_10.EPoint origin,
    Euresys.Open_eVision_2_10.EPoint middle,
    Euresys.Open_eVision_2_10.EPoint end,
    float breadth,
    bool fullCircle
)
```

Parameters

origin

Origin point coordinates of the wedge.

middle

Middle point coordinates of the wedge.

end

End point coordinates of the wedge.

breadth

Nominal breadth of the wedge. It must be negative or zero. The default value is **-50**.

fullCircle

TRUE (default) in case of a full turn wedge. If **fullCircle** is **FALSE**, **origin** and **end** give the wedge's amplitude.

Remarks

In a direct coordinate system, the abscissa extends rightwards and the ordinate extends upwards. The coordinate system is said to be inverse if the abscissa extends rightwards and the ordinate extends downwards.

EWedge.SetFromTwoPoints

DEPRECATED (you should use [EWedge::SetFromCenterAndOrigin](#)): Sets the geometric parameters (nominal position and diameter, breadth, angular position and amplitude) of an [EWedge](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetFromTwoPoints (
    Euresys.Open_eVision_2_10.EPoint center,
    Euresys.Open_eVision_2_10.EPoint origin,
    float breadth,
    bool direct
)
```

Parameters

center

Center coordinates of the wedge at its nominal position. The default value is **(0,0)**.

origin

Origin point coordinates of the wedge.

breadth

Nominal breadth of the wedge. It must be negative or zero. The default value is **-50**.

direct

TRUE (default) means that angles increase anticlockwise in a direct coordinate system and clockwise in an inverse coordinate system.

Remarks

In a direct coordinate system, the abscissa extends rightwards and the ordinate extends upwards. The coordinate system is said to be inverse if the abscissa extends rightwards and the ordinate extends downwards.

EWedge.SetRadii

Sets the nominal radius and breadth of the [EWedge](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetRadii (
    float radius,
    float breadth
)
```

Parameters

radius

Outer radius of the [EWedge](#). The default value is **50**.

breadth

Breadth of the [EWedge](#), which must be negative or zero. Its default value is **-50**.

Remarks

A [EWedge](#) is fully defined knowing its nominal position (given by the coordinates of its center), its nominal radius (diameter), its breadth, the angular position from where it extends, its angular amplitude and its outline tolerance.

By default, the [EWedge](#) radius value is **50**, which means 50 pixels when the field of view is not calibrated and 50 "units" in case of a calibrated field of view.

4.191. EWedgeGauge Class

Manages a wedge fitting gauge.

Base Class: [EWedgeShape](#)

Namespace: Euresys.Open_eVision_2_10

Properties

[Active](#)

Sets the flag indicating whether the gauge is active or not.

[ActiveEdges](#)

Active edges as defined in [EDragHandle](#).

[AverageDistance](#)

Average distance between the sampled points and the fitted model.

[FilteringThreshold](#)

Relative filtering threshold, that is the fraction of the average distance between the sampled points and the fitted model above which a point is filtered out.

HVConstraint	Status of the restriction on the orientation of the point location gauge or model fitting sample paths.
MeasuredWedge	Information pertaining to the fitted wedge.
MinAmplitude	Offset added to the Threshold when a peak is to be detected.
MinArea	Minimum area value.
NumFilteringPasses	Number of filtering passes for a model fitting operation.
NumSamples	Number of sampled points during the model fitting operation.
NumSamplesa	Number of sampled points found on edge a during the measure operation.
NumSamplesA	Number of sampled points found on edge A during the measure operation.
NumSamplesr	Number of sampled points found on edge r during the measure operation.
NumSamplesR	Number of sampled points found on edge R during the measure operation.

NumSkipRanges

Number of skip ranges in the gauge after a call to [EWedgeGauge::AddSkipRange](#).

NumValidSamples

Number of valid sample points remaining after a model fitting operation.

RectangularSamplingArea

Flag indicating whether the sampling area remains a rectangle when rotated, instead of becoming a parallelogram.

SamplingStep

Approximate distance between sampled points during a model fitting operation.

Smoothing

Number of pixels used for the low-pass filtering operation.

Thickness

Number of parallel segments used to extract the data profile.

Threshold

Threshold level used to delimit significant peaks in the data profile.

Tolerance

Searching area half thickness of the wedge fitting gauge.

TransitionChoice

Transition choice.

TransitionIndex

Index (from **0** on) of the transition to be retained when the transition choice parameter is set to [NthFromBegin](#) or [NthFromEnd](#).

TransitionType

Transition type.

Type

Shape type.

Valid

Flag indicating if at least one valid transition has been found.

Wedge

Methods

Sets the nominal position (center coordinates), diameter, breadth, angular origin and amplitude of the wedge fitting gauge according to a known wedge.

AddSkipRange

Adds an item to the set of skip ranges and returns the index of the newly added range.

CopyTo

Copies all the data of the current EWedgeGauge object into another EWedgeGauge object, and returns it.

Drag

Moves a handle to a new position and updates the position parameters of the gauge.

Draw

Draws a graphical representation of a point location or model fitting gauge, as defined by [EDrawingMode](#).

[DrawWithCurrentPen](#)

Draws a graphical representation of a point location or model fitting gauge, as defined by [EDrawingMode](#).

[EWedgeGauge](#)

Constructs a wedge measurement context.

[GetMeasuredPoint](#)

Returns the coordinates of a sample point, measured along one of the sample paths of the gauge.

[GetMinNumFitSamples](#)

Returns the minimum number of samples required for fitting on each side of the shape.

[GetSamplea](#)

Allows to retrieve information on the samples found along the a edge.

[GetSampleA](#)

Allows to retrieve information on the samples found along the A edge.

[GetSampler](#)

Allows to retrieve information on the samples found along the r edge.

[GetSampleR](#)

Allows to retrieve information on the samples found along the R edge.

[GetSkipRange](#)

Allows to retrieve the start and end values of the skip range corresponding to the given index, if it is valid (i.e. if it has previously been created by a call to the [EWedgeGauge::AddSkipRange](#) method).

HitTest

Checks whether the cursor is positioned over a handle (**TRUE**) or not (**FALSE**).

Measure

Triggers the point location or the model fitting operation.

MeasureSample

Computes the sample points along the sample path whose index in the list is given by the **pathIndex** parameter.

MeasureWithoutFitting

Triggers the point location without wedge fitting operation.

operator=

Copies all the data from another EWedgeGauge object into the current EWedgeGauge object

Plot

Draws the profile that is crossed by one of the sample paths of the gauge, as defined by [EPlotItem](#).

PlotWithCurrentPen

Draws the profile that is crossed by one of the sample paths of the gauge, as defined by [EPlotItem](#).

Process

Triggers the process pertaining to a shape or gauge and all the daughter gauges attach to it.

RemoveAllSkipRanges

Removes all the skip ranges previously created by a call to [EWedgeGauge::AddSkipRange](#).

[RemoveSkipRange](#)

After a call to [EWedgeGauge::AddSkipRange](#), removes the skip range with the given index.

[SetDiameters](#)

Sets the nominal inner/outer diameter and breadth of the [EWedgeGauge](#).

[SetFromOriginMiddleEnd](#)

Sets the geometric parameters (nominal position and diameter, breadth, angular position and amplitude) of an [EWedgeGauge](#) object.

[SetFromTwoPoints](#)

DEPRECATED (you should use [EWedgeGauge](#)): Sets the geometric parameters (nominal position and diameter, breadth, angular position and amplitude) of an [EWedgeGauge](#) object.

[SetMinNumFitSamples](#)

Sets the minimum number of samples required for fitting on each side of the shape.

[SetRadii](#)

⌈ Sets the nominal radius and breadth of the [EWedgeGauge](#).

W

edgeGauge.Active

Sets the flag indicating whether the gauge is active or not.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
override bool Active
```

```
{ get; set; }
```

Remarks

When complex gauging is required, several gauges can be grouped together. Applying [EWedgeGauge::Process](#) to the mother gauge or shape triggers the measurement of the whole. Only the active gauges will participate in the process. By default, the gauge is active (**TRUE**).

EWedgeGauge.ActiveEdges

Active edges as defined in [EDragHandle](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint ActiveEdges
```

```
{ get; set; }
```

Remarks

In the case of a wedge fitting gauge, each edge can have its own transition detection parameters. Updating the transition parameters only affect the current active edges. By default, all edges are active.

EWedgeGauge.AddSkipRange

Adds an item to the set of skip ranges and returns the index of the newly added range.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint AddSkipRange(
    uint start,
    uint end
)
```

Parameters

start

Beginning of the skip range.

end

End of the skip range.

Remarks

The samples indices between start and end (including the boundaries) will be discarded during the measurement process.

The [EWedgeGauge::AddSkipRange](#) method allows to define skip ranges in an [EWedgeGauge](#). This means that, at measure time, samples belonging to these ranges will not be taken into account.

A sample may belong to more than one skip range; to be discarded, a sample has to pertain to at least one range. Moreover, the skip ranges are allowed to overlap one another.

The range is allowed to be reversed (i.e. end is not required to be greater than start).

Also, start and end are not required to reference valid indices at the time of the call (i.e. the range may lie outside of the current return value for [EWedgeGauge::NumSamples](#)).

EWedgeGauge.AverageDistance

Average distance between the sampled points and the fitted model.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float AverageDistance
{
    get;
}
```

Remarks

Irrelevant in case of a point location operation.

EWedgeGauge.CopyTo

Copies all the data of the current EWedgeGauge object into another EWedgeGauge object, and returns it.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EWedgeGauge CopyTo(
    Euresys.Open_eVision_2_10.EWedgeGauge other,
    bool recursive
)
```

Parameters

other

Pointer to the EWedgeGauge object in which the current EWedgeGauge object data have to be copied.

recursive

TRUE if the children gauges have to be copied as well, **FALSE** otherwise.

Remarks

In case of a **NULL** pointer, a new EWedgeGauge object will be created and returned.

EWedgeGauge.Drag

Moves a handle to a new position and updates the position parameters of the gauge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void Drag(  
    int x,  
    int y  
)
```

Parameters

x

Cursor current X coordinate.

y

Cursor current Y coordinate.

EWedgeGauge.Draw

Draws a graphical representation of a point location or model fitting gauge, as defined by [EDrawingMode](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,  
    bool daughters  
)  
  
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,  
    bool daughters  
)  
  
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ERGBColor color,  
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,  
    bool daughters  
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingMode

Indicates how the point location or model fitting gauge must be displayed, as defined by [EDrawingMode](#).

daughters

TRUE if the daughters gauges are to be displayed also.

color

The color in which to draw the overlay.

EWedgeGauge.DrawWithCurrentPen

Draws a graphical representation of a point location or model fitting gauge, as defined by [EDrawingMode](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawWithCurrentPen(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingMode

Indicates how the point location or model fitting gauge must be displayed, as defined by [EDrawingMode](#).

daughters

TRUE if the daughters gauges are to be displayed also.

EWedgeGauge.EWedgeGauge

Constructs a wedge measurement context.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EWedgeGauge (
)
void EWedgeGauge (
    Euresys.Open_eVision_2_10.EWedgeGauge other
)
```

Parameters

other

Another EWedgeGauge object to be copied in the new EWedgeGauge object.

Remarks

With the default constructor, all the parameters are initialized to their respective default values. With the copy constructor, the constructed wedge measurement context is based on a pre-existing EWedgeGauge object. The gauge children are also copied. Hierarchy copying through a copy constructor is always recursive. To disable this recursion, use instead the [EWedgeGauge::CopyTo](#) method.

EWedgeGauge.FilteringThreshold

Relative filtering threshold, that is the fraction of the average distance between the sampled points and the fitted model above which a point is filtered out.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float FilteringThreshold
```

```
{ get; set; }
```

Remarks

Irrelevant in case of a point location operation. During a model fitting operation, the "filtering" process can be invoked to remove outliers, i.e. points that were located significantly far away from the fitted model (so that their position is dubious).

EWedgeGauge.GetMeasuredPoint

Returns the coordinates of a sample point, measured along one of the sample paths of the gauge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint GetMeasuredPoint(  
    uint index  
)
```

Parameters

index

This argument must be left unchanged from its default value, i.e. **~0** (= **0xFFFFFFFF**).

Remarks

These coordinates pertain to the world space; they are expressed in the reference frame to which the current EWedgeGauge object belongs. The gauging process uses a list of sample points to find the shape position and size that best fit a given image. These sample points are measured along the sample paths defined by the gauge geometry. [EWedgeGauge::GetMeasuredPoint](#) returns the coordinates of the sample point that meets the following two requirements: 1. It lies on the sample path inspected with the last call to [EWedgeGauge::MeasureSample](#), and 1. Among all the sample points along the latter sample path, it is the one selected by the [EWedgeGauge::TransitionChoice](#) property.

Note. For this method to succeed, it is necessary to previously call [EWedgeGauge::MeasureSample](#).

EWedgeGauge.GetMinNumFitSamples

Returns the minimum number of samples required for fitting on each side of the shape.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetMinNumFitSamples (
    out int side0,
    out int side1,
    out int side2,
    out int side3
)
```

Parameters

side0

Minimum number of samples on the top side of the rectangle.

side1

Minimum number of samples on the left side of the rectangle.

side2

Minimum number of samples on the bottom side of the rectangle.

side3

Minimum number of samples on the right side of the rectangle.

Remarks

Irrelevant in case of a point location operation.

EWedgeGauge.GetSampleA

Allows to retrieve information on the samples found along the A edge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool GetSampleA(
    Euresys.Open_eVision_2_10.EPoint pt,
    uint index
)

void GetSampleA(
    Euresys.Open_eVision_2_10.ESamplePoint sp,
    uint index
)

bool GetSampleA(
    ref Euresys.Open_eVision_2_10.EPeak pk,
    uint index
)
```

Parameters

pt

EPoint structure that will receive the sample position.

index

The sample index

sp

ESamplePoint structure that will receive the sample position.

pk

EPeak structure that will contain the sample peak properties.

Remarks

The method provides the sample point corresponding to the supplied index. The returned value is true when the sample is valid, and false otherwise.

EWedgeGauge.GetSampleA

Allows to retrieve information on the samples found along the A edge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool GetSampleA(
    Euresys.Open_eVision_2_10.EPoint pt,
    uint index
)

void GetSampleA(
    Euresys.Open_eVision_2_10.ESamplePoint sp,
    uint index
)

bool GetSampleA(
    ref Euresys.Open_eVision_2_10.EPeak pk,
    uint index
)
```

Parameters

pt

EPoint structure that will receive the sample position.

index

The sample index

sp

ESamplePoint structure that will receive the sample position.

pk

EPeak structure that will contain the sample peak properties.

Remarks

The method provides the sample point corresponding to the supplied index.
The returned value is true when the sample is valid, and false otherwise.

EWedgeGauge.GetSampleR

Allows to retrieve information on the samples found along the R edge.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
bool GetSampleR(
    Euresys.Open_eVision_2_10.EPoint pt,
    uint index
)

void GetSampleR(
    Euresys.Open_eVision_2_10.ESamplePoint sp,
    uint index
)

bool GetSampleR(
    ref Euresys.Open_eVision_2_10.EPeak pk,
    uint index
)
```

Parameters

pt

EPoint structure that will receive the sample position.

index

The sample index

sp

ESamplePoint structure that will receive the sample position.

pk

EPeak structure that will contain the sample peak properties.

Remarks

The method provides the sample point corresponding to the supplied index.
The returned value is true when the sample is valid, and false otherwise.

EWedgeGauge.GetSampleR

Allows to retrieve information on the samples found along the R edge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool GetSampleR(
    Euresys.Open_eVision_2_10.EPoint pt,
    uint index
)

void GetSampleR(
    Euresys.Open_eVision_2_10.ESamplePoint sp,
    uint index
)

bool GetSampleR(
    ref Euresys.Open_eVision_2_10.EPeak pk,
    uint index
)
```

Parameters

pt

[EPoint](#) structure that will receive the sample position.

index

The sample index

sp

[ESamplePoint](#) structure that will receive the sample position.

pk

[EPeak](#) structure that will contain the sample peak properties.

Remarks

The method provides the sample point corresponding to the supplied index.
The returned value is true when the sample is valid, and false otherwise.

EWedgeGauge.GetSkipRange

Allows to retrieve the start and end values of the skip range corresponding to the given index, if it is valid (i.e. if it has previously been created by a call to the [EWedgeGauge::AddSkipRange](#) method).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetSkipRange(
    uint index,
    out uint start,
    out uint end
)
```

Parameters

index

Index of the skip range.

start

Beginning of the skip range.

end

End of the skip range.

Remarks

Start is guaranteed to be smaller or equal to end.

EWedgeGauge.HitTest

Checks whether the cursor is positioned over a handle (**TRUE**) or not (**FALSE**).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool HitTest(
    bool daughters
)
```

Parameters

daughters

TRUE if the daughters gauges handles have to be considered as well.

EWedgeGauge.HVConstraint

Status of the restriction on the orientation of the point location gauge or model fitting sample paths.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool HVConstraint  
  
{ get; set; }
```

Remarks

Sample paths are the point location gauges placed along the model to be fitted.

EWedgeGauge.Measure

Triggers the point location or the model fitting operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void Measure (  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage  
)
```

Parameters

sourceImage

Pointer to the source image.

Remarks

When this method is called, and if not enough valid points are detected, then the method returns directly, and the measured gauge is set to the nominal parameters.

EWedgeGauge.MeasuredWedge

Information pertaining to the fitted wedge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EWedge MeasuredWedge  
{ get; }
```

EWedgeGauge.MeasureSample

Computes the sample points along the sample path whose index in the list is given by the **pathIndex** parameter.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void MeasureSample(  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    uint pathIndex  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

pathIndex

Sample path index.

Remarks

This method stores its results into a temporary variable inside the EWedgeGauge object.

EWedgeGauge.MeasureWithoutFitting

Triggers the point location without wedge fitting operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void MeasureWithoutFitting(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage
)
```

Parameters

sourceImage
Source image.

Remarks

This method performs the actual measurement for each transition, but does not perform the wedge fitting. This means that individual samples will be available for each edges through the [EWedgeGauge::GetSamplea](#) (Edge a), [EWedgeGauge::GetSampler](#) (Edge r), [EWedgeGauge::GetSampleA](#) (Edge A), [EWedgeGauge::GetSampleR](#) (Edge R) methods, but the gauge position will not be changed.

Please note that the filtering will not be performed in this method, since it relies upon the fitting process. The filtering parameters will thus be unused.

EWedgeGauge.MinAmplitude

Offset added to the **Threshold** when a peak is to be detected.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint MinAmplitude
```

```
{ get; set; }
```

Remarks

When analyzing a derivative profile, a peak is made up of consecutive pixel values above **Threshold**. To detect weak [strong] transitions, lower [raise] the **Threshold** value.

To avoid interference of noise, an additional parameter is provided. The **MinAmplitude** parameter is an offset added to **Threshold** when a peak is to be detected.

When the pixel values of the derivative profile do not reach **Threshold + MinAmplitude**, the peak is not taken into account. Anyway, when a peak is taken into account, all the pixels with values above **Threshold** are considered (for more accuracy). Setting the **MinAmplitude** value to **0** merely cancels its effect.

EWedgeGauge.MinArea

Minimum area value.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
uint MinArea  
  
{ get; set; }
```

Remarks

A transition is detected if its derivative peak reaches **Threshold + MinAmplitude** value, and then declared valid if the area between the peak curve and the horizontal at level **Threshold** reaches the **MinArea** value.

EWedgeGauge.NumFilteringPasses

Number of filtering passes for a model fitting operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint NumFilteringPasses
{ get; set; }
```

Remarks

Irrelevant in case of a point location operation. During a filtering pass, the points that are too distant from the model are discarded. During a model fitting operation, the "filtering" process can be invoked to remove outliers, i.e. points that were located significantly far away from the fitted model (so that their position is dubious). By default (the number of filtering passes is 0), the outliers rejection process is disabled.

EWedgeGauge.NumSamples

Number of sampled points during the model fitting operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint NumSamples
{ get; }
```

Remarks

Irrelevant in case of a point location operation.

After a model fitting operation, a number of points have been fitted along the model. Among them, some are not reliable because of their **Area** value. Among the remaining ones, some are filtered out (**NumFilteringPasses**, **FilteringThreshold**).

EWedgeGauge.NumSamplesA

Number of sampled points found on edge A during the measure operation.

Namespace: Euresys.Open_eVision_2_10


```
[C#]  
uint NumSamplesA  
  
    { get; }
```

EWedgeGauge.NumSamplesA

Number of sampled points found on edge A during the measure operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint NumSamplesA  
  
    { get; }
```

EWedgeGauge.NumSamplesR

Number of sampled points found on edge R during the measure operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint NumSamplesR  
  
    { get; }
```

EWedgeGauge.NumSamplesR

Number of sampled points found on edge R during the measure operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint NumSamplesR  
    { get; }
```

EWedgeGauge.NumSkipRanges

Number of skip ranges in the gauge after a call to [EWedgeGauge::AddSkipRange](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint NumSkipRanges  
    { get; }
```

EWedgeGauge.NumValidSamples

Number of valid sample points remaining after a model fitting operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint NumValidSamples
{ get; }
```

Remarks

Irrelevant in case of a point location operation. After a model fitting operation, a number of points have been fitted along the model. Among them, some are not reliable because of their **Area** value. Among the remaining ones, some are filtered out (**NumFilteringPasses**, **FilteringThreshold**).

EWedgeGauge.operator=

Copies all the data from another EWedgeGauge object into the current EWedgeGauge object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EWedgeGauge operator=(
    Euresys.Open_eVision_2_10.EWedgeGauge other
)
```

Parameters

other
EWedgeGauge object to be copied

EWedgeGauge.Plot

Draws the profile that is crossed by one of the sample paths of the gauge, as defined by [EPlotItem](#).

Namespace: Euresys.Open_eVision_2_10

```

[C#]
void Plot(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.EPlotItem drawItems,
    float width,
    float height,
    float originX,
    float originY
)

void Plot(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EPlotItem drawItems,
    float width,
    float height,
    float originX,
    float originY
)

void Plot(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.EPlotItem drawItems,
    float width,
    float height,
    float originX,
    float originY
)

```

Parameters

graphicContext

Handle of the device context on which to draw.

drawItems

Boolean combination of [EPlotItem](#) members, that indicates which items must be displayed.

width

Width of the plot.

height

Height of the plot.

originX

Origin point coordinates of the plot along the X axis.

originY

Origin point coordinates of the plot along the Y axis.

color

The color in which to draw the overlay.

Remarks

The sample path that is taken into considered is the one inspected with the last call to [EWedgeGauge::MeasureSample](#).

Note. For this method to succeed, it is necessary to previously call [EWedgeGauge::MeasureSample](#).

EWedgeGauge.PlotWithCurrentPen

Draws the profile that is crossed by one of the sample paths of the gauge, as defined by [EPlotItem](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void PlotWithCurrentPen(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EPlotItem drawItems,
    float width,
    float height,
    float originX,
    float originY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawItems

Boolean combination of [EPlotItem](#) members, that indicates which items must be displayed.

width

Width of the plot.

height

Height of the plot.

originX

Origin point coordinates of the plot along the X axis.

originY

Origin point coordinates of the plot along the Y axis.

Remarks

The sample path that is taken into considered is the one inspected with the last call to [EWedgeGauge::MeasureSample](#).

Note. For this method to succeed, it is necessary to previously call [EWedgeGauge::MeasureSample](#).

EWedgeGauge.Process

Triggers the process pertaining to a shape or gauge and all the daughter gauges attach to it.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Process (
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    bool daughters
)
```

Parameters

sourceImage

Pointer to the source image.

daughters

Flag indicating whether the daughters shapes inherit of the same behavior.

Remarks

When complex gauging is required, several gauges can be grouped together. Applying **Process** to the mother gauge or shape triggers the measurement of the whole. Only the active gauges will participate in the process.

EWedgeGauge.RectangularSamplingArea

Flag indicating whether the sampling area remains a rectangle when rotated, instead of becoming a parallelogram.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
bool RectangularSamplingArea  
  
{ get; set; }
```

Remarks

By default, this flag is set to **TRUE**: the sampling area always remains a rectangle. Setting this property is only useful when the thickness transition parameter is greater than 1. In fact, when thickness transition parameter is equal to 1, rectangle and parallelogram reduce to the same segment.

EWedgeGauge.RemoveAllSkipRanges

Removes all the skip ranges previously created by a call to [EWedgeGauge::AddSkipRange](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void RemoveAllSkipRanges (  
)
```

EWedgeGauge.RemoveSkipRange

After a call to [EWedgeGauge::AddSkipRange](#), removes the skip range with the given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void RemoveSkipRange (  
    uint index  
)
```

Parameters

index

Index of the skip range to remove, as returned by [EWedgeGauge::AddSkipRange](#).

EWedgeGauge.SamplingStep

Approximate distance between sampled points during a model fitting operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
float SamplingStep  
  
    { get; set; }
```

Remarks

Irrelevant in case of a point location operation.

To fit a model, a series of point location operations are performed along the model. The point location gauges spacing is given by the sampling step.

By default, the sampling step is set to **5**, which means 5 pixels when the field of view is not calibrated, and 5 physical units in case of a calibrated field of view.

Be aware that if the sampling step is too large, the number of sampled points along the model will not be sufficient enough to accurately locate it.

EWedgeGauge.SetDiameters

Sets the nominal inner/outer diameter and breadth of the [EWedgeGauge](#).

Namespace: Euresys.Open_eVision_2_10


```
[C#]
void SetDiameters (
    float diameter,
    float breadth
)
```

Parameters

diameter

Outer diameter of the [EWedgeGauge](#). The default value is **100**.

breadth

Breadth of the [EWedgeGauge](#). It must be negative or zero. Its default value is **-50**.

Remarks

A [EWedgeGauge](#) is fully defined knowing its nominal position (given by the coordinates of its center), its nominal outer radius (diameter), its breadth, the angular position from where it extends, its angular amplitude and its outline tolerance.

By default, the [EWedgeGauge](#) diameter value is **100**, which means 100 pixels when the field of view is not calibrated and 100 "units" in case of a calibrated field of view.

EWedgeGauge.SetFromOriginMiddleEnd

Sets the geometric parameters (nominal position and diameter, breadth, angular position and amplitude) of an [EWedgeGauge](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetFromOriginMiddleEnd (
    Euresys.Open_eVision_2_10.EPoint origin,
    Euresys.Open_eVision_2_10.EPoint middle,
    Euresys.Open_eVision_2_10.EPoint end,
    float breadth,
    bool full
)
```

Parameters

origin

Origin point coordinates of the wedge.

middle

Middle point coordinates of the wedge.

end

End point coordinates of the wedge.

breadth

Nominal breadth of the wedge. It must be negative or zero. The default value is **-50**.

full

-

Remarks

In a direct coordinate system, the abscissa extends rightwards and the ordinate extends upwards. The coordinate system is said to be inverse if the abscissa extends rightwards and the ordinate extends downwards.

EWedgeGauge.SetFromTwoPoints

DEPRECATED (you should use [EWedgeGauge](#)): Sets the geometric parameters (nominal position and diameter, breadth, angular position and amplitude) of an [EWedgeGauge](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetFromTwoPoints(
    Euresys.Open_eVision_2_10.EPoint center,
    Euresys.Open_eVision_2_10.EPoint origin,
    float breadth,
    bool direct
)
```

Parameters

center

Center coordinates of the wedge at its nominal position. The default value is **(0,0)**.

origin

Origin point coordinates of the wedge.

breadth

Nominal breadth of the wedge. It must be negative or zero. The default value is **-50**.

direct

TRUE (default) means that angles increase anticlockwise in a direct coordinate system and clockwise in an inverse coordinate system.

Remarks

In a direct coordinate system, the abscissa extends rightwards and the ordinate extends upwards. The coordinate system is said to be inverse if the abscissa extends rightwards and the ordinate extends downwards.

EWedgeGauge.SetMinNumFitSamples

Sets the minimum number of samples required for fitting on each side of the shape.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetMinNumFitSamples (
    int side0,
    int side1,
    int side2,
    int side3
)
```

Parameters

side0

Minimum number of samples on the *outer circle* of the wedge. The default value is **3**.

side1

Minimum number of samples on the *original border* of the wedge. If this value is not specified, it is equal to **n32Side0**. The default value is **2**.

side2

Minimum number of samples on the *inner circle* of the wedge. If this value is not specified, it is equal to **n32Side0**. The default value is **3**.

side3

Minimum number of samples on the *end border* of the wedge. If this value is not specified, it is equal to **n32Side1**. The default value is **2**.

Remarks

Irrelevant in case of a point location operation. When the [EWedgeGauge::Measure](#) method is called, and if not enough valid points are detected, then the method returns directly, and the measured gauge is set to the nominal parameters.

EWedgeGauge.SetRadii

Sets the nominal radius and breadth of the [EWedgeGauge](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetRadii(
    float outerRadius,
    float breadth
)
```

Parameters

outerRadius

-

breadth

Breadth of the [EWedgeGauge](#), which must be negative or zero. Its default value is **-50**.

Remarks

A [EWedgeGauge](#) is fully defined knowing its nominal position (given by the coordinates of its center), its nominal radius (diameter), its breadth, the angular position from where it extents, its angular amplitude and its outline tolerance.

By default, the [EWedgeGauge](#) radius value is **50**, which means 50 pixels when the field of view is not calibrated and 50 "units" in case of a calibrated field of view.

EWedgeGauge.Smoothing

Number of pixels used for the low-pass filtering operation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint Smoothing
```

```
{ get; set; }
```

Remarks

To reduce the effect of noise, the profile data can be low-pass filtered along the point location gauge direction.

EWedgeGauge.Thickness

Number of parallel segments used to extract the data profile.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint Thickness
```

```
{ get; set; }
```

Remarks

To reduce the effect of noise and/or strengthen a transition, several parallel profiles can be accumulated.

EWedgeGauge.Threshold

Threshold level used to delimit significant peaks in the data profile.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint Threshold
```

```
{ get; set; }
```

Remarks

When analyzing a derivative profile, a peak is made up of consecutive pixel values above **Threshold**. To detect weak [strong] transitions, lower [raise] the **Threshold** value.

To avoid interference of noise, an additional parameter is provided. The **MinAmplitude** parameter is an offset added to **Threshold** when a peak is to be detected.

When the pixel values of the derivative profile do not reach **Threshold + MinAmplitude**, the peak is not taken into account. Anyway, when a peak is taken into account, all the pixels with values above **Threshold** are considered (for more accuracy). Setting the **MinAmplitude** value to **0** merely cancels its effect.

EWedgeGauge.Tolerance

Searching area half thickness of the wedge fitting gauge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float Tolerance
```

```
{ get; set; }
```

Remarks

A wedge fitting gauge is fully defined knowing its nominal position (its center coordinates), its outer nominal radius (diameter), its breadth, the angular position from where it extends, its angular amplitude, and its outline tolerance.

By default, the searching area thickness of the wedge fitting gauge is **20** (2x10), which means 20 pixels when the field of view is not calibrated, and 20 physical units in case of a calibrated field of view.

EWedgeGauge.TransitionChoice

Transition choice.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.ETransitionChoice TransitionChoice  
  
{ get; set; }
```

Remarks

Several peaks may be detected along a point location gauge. This parameter helps to select the desired transition. In case of [NthFromBegin](#) or [NthFromEnd](#) transition choice, set [EWedgeGauge::TransitionIndex](#) to specify the desired transition.

By default, the selected transition corresponds to the one with the largest amplitude ([LargestAmplitude](#)).

EWedgeGauge.TransitionIndex

Index (from **0** on) of the transition to be retained when the transition choice parameter is set to [NthFromBegin](#) or [NthFromEnd](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint TransitionIndex  
  
{ get; set; }
```

Remarks

Several peaks may be detected along a point location gauge. This parameter helps to select the desired transition. By default, the first transition is retained (the index value is **0**).

EWedgeGauge.TransitionType

Transition type.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.ETransitionType TransitionType  
  
{ get; set; }
```

Remarks

The type of a transition tells whether it crosses increasing or decreasing gray-level values. This helps discriminate between nearby edges of an object.

By default, the searched transition type is indifferently a black to white or a white to black transition ([BwOrWb](#)).

EWedgeGauge.Type

Shape type.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
override Euresys.Open_eVision_2_10.EShapeType Type  
  
{ get; }
```

EWedgeGauge.Valid

Flag indicating if at least one valid transition has been found.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```



```
bool Valid
```

```
{ get; }
```

Remarks

A **FALSE** value means that no measurement has been performed. A **TRUE** value means that a transition was found along the sample path inspected with the last call to [EWedgeGauge::MeasureSample](#), and thus a point has been measured.

EWedgeGauge.Wedge

Sets the nominal position (center coordinates), diameter, breadth, angular origin and amplitude of the wedge fitting gauge according to a known wedge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
override Euresys.Open_eVision_2_10.EWedge Wedge
```

```
{ get; set; }
```

4.192. EWedgeShape Class

Manages a wedge shape.

Base Class: [EShape](#)

Derived Class(es): [EWedgeGauge](#)

Namespace: Euresys.Open_eVision_2_10

Properties

Amplitude	Angular amplitude of the EWedgeShape .
Angle	Orientation of the shape.
ApexAngle	Angular position at the apex of the EWedgeShape .
Breadth	Breadth of the EWedgeShape .
Center	Center point of the shape.
CenterX	Abscissa of the origin point of the shape.
CenterY	Ordinate of the origin point of the shape.
Direct	Flag indicating whether positive angles correspond to a clockwise or an anticlockwise rotation.
EndAngle	Ending angular position of the EWedgeShape .
FullBreadth	Flag indicating if the EWedgeShape has a full breadth (breadth = radius).

FullCircle	Flag indicating if the EWedgeShape is a full circle (amplitude = 2 PI).
InnerApex	Inner apex point coordinates of the EWedgeShape .
InnerArcLength	Inner circle arc length of the EWedgeShape .
InnerDiameter	Inner diameter of the EWedgeShape .
InnerEnd	Inner end point coordinates of the EWedgeShape .
InnerOrg	Inner origin point coordinates of the EWedgeShape .
InnerRadius	Inner radius of the EWedgeShape
OrgAngle	Angular position from where the EWedgeShape extends.
OuterApex	Outer apex point coordinates of the EWedgeShape .
OuterArcLength	Outer circle arc length of the EWedgeShape .
OuterDiameter	Outer diameter of the EWedgeShape .
OuterEnd	Outer end point coordinates of the EWedgeShape .

OuterOrg

Outer origin point coordinates of the [EWedgeShape](#).

OuterRadius

Outer radius of the [EWedgeShape](#).

Scale

Horizontal sensor resolution, in pixels per unit.

Type

Shape type.

Wedge

M Sets the nominal position, length and rotation angle of the wedge according to a known [EWedge](#) object.

e

Methods

Closest

Find the daughter shape that is the closest to this shape. To retrieve the closest shape, use [EShape::ClosestShape](#).

CopyTo

Copies all the data of the current [EWedgeShape](#) object into another [EWedgeShape](#) object and returns it.

Drag

Moves a handle to a new position and updates the position parameters of the shape.

Draw

Draws a graphical representation of a shape, as defined by [EDrawingMode](#).

DrawWithCurrentPen

Draws a graphical representation of a shape, as defined by [EDrawingMode](#).

`EWedgeShape`

Constructs a `EWedgeShape` object.

`GetCorners`

Retrieves the coordinates of each corner of the `EWedgeShape`.

`GetEdges`

Retrieves each edge of the `EWedgeShape`.

`GetInnerPoint`

Returns the coordinates of a particular point specified by its location along the inner circle arc.

`GetMidEdges`

Retrieves the center coordinates of each edge of the wedge fitting gauge.

`GetOuterPoint`

Returns the coordinates of a particular point specified by its location along the outer circle arc.

`GetPoint`

Returns the coordinates of a particular point specified by its location along the wedge perimeter.

`HitTest`

Checks if there is a handle under the cursor.

`operator=`

Copies all the data from another `EWedgeShape` object into the current `EWedgeShape` object

`SetCenterXY`

Sets the center coordinates of a `EWedgeShape` object.

[SetDiameters](#)

Sets the nominal inner/outer diameter and breadth of the [EWedgeShape](#).

[SetFromCenterAndOrigin](#)

Sets the geometric parameters (nominal position and diameter, breadth, angular position and amplitude) of an [EWedgeShape](#) object.

[SetFromOriginMiddleEnd](#)

Sets the geometric parameters (nominal position and diameter, breadth, angular position and amplitude) of an [EWedgeShape](#) object.

[SetFromTwoPoints](#)

DEPRECATED (you should use [EWedgeShape::SetFromCenterAndOrigin](#)): Sets the geometric parameters (nominal position and diameter, breadth, angular position and amplitude) of an [EWedgeShape](#) object.

[SetRadii](#)

⌈ Sets the nominal radius and breadth of the [EWedgeShape](#).

W

edgeShape.Amplitude

Angular amplitude of the [EWedgeShape](#).

Namespace: Euresys.Open_eVision_2_10

[C#]

```
float Amplitude
```

```
{ get; set; }
```

Remarks

A [EWedgeShape](#) is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), its breadth, the angular position from where it extends, its angular amplitude, and its outline tolerance.

The sign of the rotation angle depends whether the field of view is calibrated or not.

* When the field of view is calibrated, the coordinate system is said to be direct, the abscissa extends rightwards and the ordinate extends upwards. In this case, an anticlockwise rotation leads to a positive angle value.

* When the field of view is not calibrated, the coordinate system is said to be inverse, the abscissa extends rightwards and the ordinate extends downwards. In this case, a clockwise rotation leads to a positive angle value.

EWedgeShape.Angle

Orientation of the shape.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float Angle
```

```
{ get; set; }
```

EWedgeShape.ApexAngle

Angular position at the apex of the [EWedgeShape](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ApexAngle
```

```
{ get; }
```

Remarks

A [EWedgeShape](#) is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), its breadth, the angular position from where it extents, its angular amplitude, and its outline tolerance.

The sign of the rotation angle depends whether the field of view is calibrated or not.

* When the field of view is calibrated, the coordinate system is said to be direct, the abscissa extends rightwards and the ordinate extends upwards. In this case, an anticlockwise rotation leads to a positive angle value.

* When the field of view is not calibrated, the coordinate system is said to be inverse, the abscissa extends rightwards and the ordinate extends downwards. In this case, a clockwise rotation leads to a positive angle value.

EWedgeShape.Breadth

Breadth of the [EWedgeShape](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float Breadth
```

```
{ get; }
```

Remarks

A [EWedgeShape](#) is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), its breadth, the angular position from where it extents, its angular amplitude, and its outline tolerance.

EWedgeShape.Center

Center point of the shape.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
```

```
Euresys.Open_eVision_2_10.EPoint Center  
{ get; set; }
```

EWedgeShape.CenterX

Abscissa of the origin point of the shape.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float CenterX  
{ get; }
```

EWedgeShape.CenterY

Ordinate of the origin point of the shape.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float CenterY  
{ get; }
```

EWedgeShape.Closest

Find the daughter shape that is the closest to this shape. To retrieve the closest shape, use [EShape::ClosestShape](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Closest(
)
```

EWedgeShape.CopyTo

Copies all the data of the current [EWedgeShape](#) object into another [EWedgeShape](#) object and returns it.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EWedgeShape CopyTo(
    Euresys.Open_eVision_2_10.EWedgeShape dest,
    bool bRecursive
)
```

Parameters

dest

Pointer to the [EWedgeShape](#) object in which the current [EWedgeShape](#) object data have to be copied.

bRecursive

TRUE if the children shapes have to be copied as well, **FALSE** otherwise.

Remarks

In case of a **NULL** pointer, a new [EWedgeShape](#) object will be created and returned.

EWedgeShape.Direct

Flag indicating whether positive angles correspond to a clockwise or an anticlockwise rotation.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool Direct
{ get; }
```

Remarks

TRUE (default) means that angles increase anticlockwisely in a direct coordinate system, and clockwisely in an inverse coordinate system.

* When the field of view is calibrated, the coordinate system is said to be direct, the abscissa extends rightwards and the ordinate extends upwards.

* When the field of view is not calibrated, the coordinate system is said to be inverse the abscissa extends rightwards and the ordinate extends downwards.

EWedgeShape.Drag

Moves a handle to a new position and updates the position parameters of the shape.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Drag(
    int cursorX,
    int cursorY
)
```

Parameters

cursorX

-

cursorY

-

EWedgeShape.Draw

Draws a graphical representation of a shape, as defined by [EDrawingMode](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingMode

Indicates how the point location or model fitting gauge must be displayed, as defined by [EDrawingMode](#).

daughters

TRUE if the daughters gauges are to be displayed also.

color

-

EWedgeShape.DrawWithCurrentPen

Draws a graphical representation of a shape, as defined by [EDrawingMode](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawWithCurrentPen(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingMode,
    bool daughters
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingMode

Indicates how the point location or model fitting gauge must be displayed, as defined by [EDrawingMode](#).

daughters

TRUE if the daughters gauges are to be displayed also.

EWedgeShape.EndAngle

Ending angular position of the [EWedgeShape](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float EndAngle
{ get; }
```

Remarks

A [EWedgeShape](#) is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), its breadth, the angular position from where it extends, its angular amplitude, and its outline tolerance.

The sign of the rotation angle depends whether the field of view is calibrated or not.

* When the field of view is calibrated, the coordinate system is said to be direct, the abscissa extends rightwards and the ordinate extends upwards. In this case, an anticlockwise rotation leads to a positive angle value.

* When the field of view is not calibrated, the coordinate system is said to be inverse, the abscissa extends rightwards and the ordinate extends downwards. In this case, a clockwise rotation leads to a positive angle value.

EWedgeShape.EWedgeShape

Constructs a [EWedgeShape](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EWedgeShape (
)
void EWedgeShape (
    Euresys.Open_eVision_2_10.EWedgeShape other
)
```

Parameters

other

Another [EWedgeShape](#) object to be copied in the new [EWedgeShape](#) object.

EWedgeShape.FullBreadth

Flag indicating if the [EWedgeShape](#) has a full breadth (**breadth = radius**).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool FullBreadth  
    { get; }
```

EWedgeShape.FullCircle

Flag indicating if the [EWedgeShape](#) is a full circle (**amplitude = 2 PI**).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool FullCircle  
    { get; }
```

EWedgeShape.GetCorners

Retrieves the coordinates of each corner of the [EWedgeShape](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetCorners (
    Euresys.Open_eVision_2_10.EPoint ar,
    Euresys.Open_eVision_2_10.EPoint AAr,
    Euresys.Open_eVision_2_10.EPoint aRR,
    Euresys.Open_eVision_2_10.EPoint AARR
)
```

Parameters

ar

Coordinates of the inner org corner of the [EWedgeShape](#).

AAr

Coordinates of the inner end corner of the [EWedgeShape](#).

aRR

Coordinates of the outer org corner of the [EWedgeShape](#).

AARR

Coordinates of the outer end corner of the [EWedgeShape](#).

EWedgeShape.GetEdges

Retrieves each edge of the [EWedgeShape](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void GetEdges (
    Euresys.Open_eVision_2_10.ELine a,
    Euresys.Open_eVision_2_10.ELine AA,
    Euresys.Open_eVision_2_10.ECircle r,
    Euresys.Open_eVision_2_10.ECircle RR
)
```

Parameters

a

Org edge of the [EWedgeShape](#).

AA

End edge of the [EWedgeShape](#).

r

Inner edge of the [EWedgeShape](#).

RR

Outer edge of the [EWedgeShape](#).

EWedgeShape.GetInnerPoint

Returns the coordinates of a particular point specified by its location along the inner circle arc.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint GetInnerPoint(
    float fraction
)
```

Parameters

fraction

Point location expressed as a fraction of the circle arc (range **[-1, +1]**).

EWedgeShape.GetMidEdges

Retrieves the center coordinates of each edge of the wedge fitting gauge.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void GetMidEdges (
    Euresys.Open_eVision_2_10.EPoint a,
    Euresys.Open_eVision_2_10.EPoint AA,
    Euresys.Open_eVision_2_10.EPoint r,
    Euresys.Open_eVision_2_10.EPoint RR
)
```

Parameters

a

Center coordinates of the org edge of the [EWedgeShape](#).

AA

Center coordinates of the end edge of the [EWedgeShape](#).

r

Center coordinates of the inner edge of the [EWedgeShape](#).

RR

Center coordinates of the outer edge of the [EWedgeShape](#).

EWedgeShape.GetOuterPoint

Returns the coordinates of a particular point specified by its location along the outter circle arc.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint GetOuterPoint(
    float fraction
)
```

Parameters

fraction

Point location expressed as a fraction of the circle arc (range **[-1, +1]**).

EWedgeShape.GetPoint

Returns the coordinates of a particular point specified by its location along the wedge perimeter.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint GetPoint(
    float breadthFraction,
    float angleFraction
)
```

Parameters

breadthFraction

Point location expressed as a fraction of the wedge breadth (range -1, +1).

angleFraction

Point location expressed as a fraction of the wedge amplitude (range -1, +1).

EWedgeShape.HitTest

Checks if there is a handle under the cursor.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool HitTest(
    bool daughters
)
```

Parameters

daughters

EWedgeShape.InnerApex

Inner apex point coordinates of the [EWedgeShape](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint InnerApex  
{ get; }
```

EWedgeShape.InnerArcLength

Inner circle arc length of the [EWedgeShape](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float InnerArcLength  
{ get; }
```

EWedgeShape.InnerDiameter

Inner diameter of the [EWedgeShape](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float InnerDiameter  
  
    { get; }
```

EWedgeShape.InnerEnd

Inner end point coordinates of the [EWedgeShape](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint InnerEnd  
  
    { get; }
```

EWedgeShape.InnerOrg

Inner origin point coordinates of the [EWedgeShape](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint InnerOrg  
  
    { get; }
```

EWedgeShape.InnerRadius

Inner radius of the [EWedgeShape](#)

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float InnerRadius  
  
    { get; }
```

EWedgeShape.operator=

Copies all the data from another EWedgeShape object into the current EWedgeShape object

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EWedgeShape operator=(  
    Euresys.Open_eVision_2_10.EWedgeShape other  
)
```

Parameters

other
EWedgeShape object to be copied

EWedgeShape.OrgAngle

Angular position from where the [EWedgeShape](#) extents.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float OrgAngle  
  
    { get; }
```

Remarks

A [EWedgeShape](#) is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), its breadth, the angular position from where it extents, its angular amplitude, and its outline tolerance.

The sign of the rotation angle depends whether the field of view is calibrated or not.

* When the field of view is calibrated, the coordinate system is said to be direct, the abscissa extends rightwards and the ordinate extends upwards. In this case, an anticlockwise rotation leads to a positive angle value.

* When the field of view is not calibrated, the coordinate system is said to be inverse, the abscissa extends rightwards and the ordinate extends downwards. In this case, a clockwise rotation leads to a positive angle value.

EWedgeShape.OuterApex

Outer apex point coordinates of the [EWedgeShape](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint OuterApex  
  
    { get; }
```

Remarks

A [EWedgeShape](#) is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), its breadth, the angular position from where it extents, its angular amplitude, and its outline tolerance.

EWedgeShape.OuterArcLength

Outer circle arc length of the [EWedgeShape](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float OuterArcLength  
  
    { get; }
```

Remarks

A [EWedgeShape](#) is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), its breadth, the angular position from where it extends, its angular amplitude, and its outline tolerance.

EWedgeShape.OuterDiameter

Outer diameter of the [EWedgeShape](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float OuterDiameter  
  
    { get; }
```

Remarks

A [EWedgeShape](#) is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), its breadth, the angular position from where it extends, its angular amplitude, and its outline tolerance.

EWedgeShape.OuterEnd

Outer end point coordinates of the [EWedgeShape](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint OuterEnd  
  
{ get; }
```

Remarks

A [EWedgeShape](#) is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), its breadth, the angular position from where it extents, its angular amplitude, and its outline tolerance.

EWedgeShape.OuterOrg

Outer origin point coordinates of the [EWedgeShape](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EPoint OuterOrg  
  
{ get; }
```

Remarks

A [EWedgeShape](#) is fully defined knowing its nominal position (its center coordinates), its nominal outer radius (diameter), its breadth (must be negative), the angular position from where it extents, its angular amplitude, and its outline tolerance.

EWedgeShape.OuterRadius

Outer radius of the [EWedgeShape](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float OuterRadius  
  
    { get; }
```

Remarks

A [EWedgeShape](#) is fully defined knowing its nominal position (its center coordinates), its nominal radius (diameter), its breadth, the angular position from where it extents, its angular amplitude, and its outline tolerance.

EWedgeShape.Scale

Horizontal sensor resolution, in pixels per unit.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Scale  
  
    { get; set; }
```

EWedgeShape.SetCenterXY

Sets the center coordinates of a [EWedgeShape](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetCenterXY(
    float centerX,
    float centerY
)
```

Parameters

centerX

Center coordinates of the [EWedgeShape](#) object.

centerY

Center coordinates of the [EWedgeShape](#) object.

EWedgeShape.SetDiameters

Sets the nominal inner/outer diameter and breadth of the [EWedgeShape](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetDiameters(
    float diameter,
    float breadth
)
```

Parameters

diameter

Outer diameter of the [EWedgeShape](#). The default value is **100**.

breadth

Breadth of the [EWedgeShape](#). It must be negative or zero. Its default value is **-50**.

Remarks

A [EWedgeShape](#) is fully defined knowing its nominal position (given by the coordinates of its center), its nominal outer radius (diameter), its breadth, the angular position from where it extends, its angular amplitude and its outline tolerance.

By default, the [EWedgeShape](#) diameter value is **100**, which means 100 pixels when the field of view is not calibrated and 100 "units" in case of a calibrated field of view.

EWedgeShape.SetFromCenterAndOrigin

Sets the geometric parameters (nominal position and diameter, breadth, angular position and amplitude) of an [EWedgeShape](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetFromCenterAndOrigin(
    Euresys.Open_eVision_2_10.EPoint center,
    Euresys.Open_eVision_2_10.EPoint origin,
    float breadth,
    bool direct
)
```

Parameters

center

Center coordinates of the wedge at its nominal position. The default value is **(0,0)**.

origin

Origin point coordinates of the wedge.

breadth

Nominal breadth of the wedge. It must be negative or zero. The default value is **-50**.

direct

TRUE (default) means that angles increase anticlockwise in a direct coordinate system and clockwise in an inverse coordinate system.

Remarks

In a direct coordinate system, the abscissa extends rightwards and the ordinate extends upwards. The coordinate system is said to be inverse if the abscissa extends rightwards and the ordinate extends downwards.

EWedgeShape.SetFromOriginMiddleEnd

Sets the geometric parameters (nominal position and diameter, breadth, angular position and amplitude) of an [EWedgeShape](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetFromOriginMiddleEnd(
    Euresys.Open_eVision_2_10.EPoint origin,
    Euresys.Open_eVision_2_10.EPoint middle,
    Euresys.Open_eVision_2_10.EPoint end,
    float breadth,
    bool fullCircle
)
```

Parameters

origin

Origin point coordinates of the wedge.

middle

Middle point coordinates of the wedge.

end

End point coordinates of the wedge.

breadth

Nominal breadth of the wedge. It must be negative or zero. The default value is **-50**.

fullCircle

TRUE (default) in case of a full turn wedge. If **fullCircle** is **FALSE**, **origin** and **end** give the wedge's amplitude.

Remarks

In a direct coordinate system, the abscissa extends rightwards and the ordinate extends upwards. The coordinate system is said to be inverse if the abscissa extends rightwards and the ordinate extends downwards.

EWedgeShape.SetFromTwoPoints

DEPRECATED (you should use [EWedgeShape::SetFromCenterAndOrigin](#)): Sets the geometric parameters (nominal position and diameter, breadth, angular position and amplitude) of an [EWedgeShape](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetFromTwoPoints(
    Euresys.Open_eVision_2_10.EPoint center,
    Euresys.Open_eVision_2_10.EPoint origin,
    float breadth,
    bool direct
)
```

Parameters

center

Center coordinates of the wedge at its nominal position. The default value is **(0,0)**.

origin

Origin point coordinates of the wedge.

breadth

Nominal breadth of the wedge. It must be negative or zero. The default value is **-50**.

direct

TRUE (default) means that angles increase anticlockwise in a direct coordinate system and clockwise in an inverse coordinate system.

Remarks

In a direct coordinate system, the abscissa extends rightwards and the ordinate extends upwards. The coordinate system is said to be inverse if the abscissa extends rightwards and the ordinate extends downwards.

EWedgeShape.SetRadii

Sets the nominal radius and breadth of the [EWedgeShape](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetRadii(
    float outerRadius,
    float breadth
)
```

Parameters

outerRadius

Outer radius of the [EWedgeShape](#). The default value is **50**.

breadth

Breadth of the [EWedgeShape](#), which must be negative or zero. Its default value is **-50**.

Remarks

A [EWedgeShape](#) is fully defined knowing its nominal position (given by the coordinates of its center), its nominal radius (diameter), its breadth, the angular position from where it extents, its angular amplitude and its outline tolerance.

By default, the [EWedgeShape](#) radius value is **50**, which means 50 pixels when the field of view is not calibrated and 50 "units" in case of a calibrated field of view.

EWedgeShape.Type

Shape type.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
override Euresys.Open_eVision_2_10.EShapeType Type
{ get; }
```

EWedgeShape.Wedge

Sets the nominal position, length and rotation angle of the wedge according to a known [EWedge](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
virtual Euresys.Open_eVision_2_10.EWedge Wedge
    { get; set; }
```

4.193. EWorldShape Class

Manages a complete context for calibrating a field of view.

Base Class: [EShape](#)

Namespace: Euresys.Open_eVision_2_10

Properties

Angle	Tilt X angle, that is the amplitude of the rotation applied around the X-axis of the sensor to bring the optical (Z) axis perpendicular to the field of view.
CalibrationModes	Current calibration mode, made from a combination of values.
Center	Position of the origin point of the reference frame as projected onto the image, i.e. the Sensor position of the point at World coordinates (0,0) .
CenterX	Horizontal position of the origin point of the reference frame as projected onto the image, i.e. the Sensor abscissa of the point at World coordinates (0,0) .

CenterY

Vertical position of the origin point of the reference frame as projected onto the image, i.e. the Sensor ordinate of the point at World coordinates **(0,0)**.

DistortionStrength

Optical distortion strength, i.e. the ratio of the image diagonal length with and without optical distortion introduced by the lens.

DistortionStrength2

Optical distortion strength of the second order.

FieldHeight

Field-of-view height, in physical units.

FieldWidth

Field-of-view width, in physical units.

GridPointsMaxVariation

Maximum variation of grid points (maximum distance between computed grid points and ideal grid points in world space), normalized using the pitch values, after a call to [EWorldShape::CalibrationSucceeded](#)

GridPointsMaxVariationThreshold

Grid points maximum variation threshold, that is the value above which the maximum variation of grid points (maximum distance between computed grid points and ideal grid points in world space) is considered too high, and thus marks the calibration as a failure when using [EWorldShape::CalibrationSucceeded](#).

GridPointsMeanVariation

Mean variation of grid points (mean distance between computed grid points and ideal grid points in world space), normalized using the pitch values, after a call to [EWorldShape::CalibrationSucceeded](#).

GridPointsMeanVariationThreshold

Grid points mean variation threshold, that is the value above which the mean variation of grid points (mean distance between computed grid points and ideal grid points in world space) is considered too high, and thus marks the calibration as a failure when using [EWorldShape::CalibrationSucceeded](#).

HitLandmark

Returns the landmark selected by [EWorldShape::HitLandmarks](#) or ~0 if no landmark is selected.

NumLandmarkElements

Returns the number of landmark elements.

PanX

Current horizontal panning factor for drawing operations.

PanY

Current vertical panning factor for drawing operations.

PerspectiveStrength

Perspective effect coefficient, that is the inverse of the observation distance.

Ratio

XResolution/YResolution ratio.

Scale

The scale of the frame.

SensorHeight

Logical image height, that is the number of pixels vertically.

SensorWidth

Logical image width, that is the number of pixels horizontally.

TiltXAngle

Tilt X angle, that is the amplitude of the rotation applied around the X-axis of the sensor to bring the optical (Z) axis perpendicular to the field of view.

TiltYAngle

Tilt Y angle, that is the amplitude of the rotation applied around the Y-axis of the sensor to bring the optical (Z) axis perpendicular to the field of view.

Type

Shape type.

XResolution

Horizontal sensor resolution, in pixels per unit.

YResolution

Vertical sensor resolution, in pixels per unit.

ZoomX

Current horizontal zooming factor for drawing operations.

ZoomY

M Current vertical zooming factor for drawing operations.

e

thods

AddLandmark

Adds a new pair of points coordinates (in Sensor and World spaces) to the set of landmarks used for calibration.

AddPoint

Adds a new point coordinates (in Sensor space) to the set of grid points used for calibration.

AutoCalibrate

Returns the best calibration modes for the current calibration grid and calibrates the field of view accordingly.

AutoCalibrateDotGrid

Performs an automatic calibration based on a dot grid image.

AutoCalibrateLandmarks

Returns the best calibration modes for the current landmark set and calibrates the field of view accordingly.

Calibrate

Performs a calibration according to the specified combination of calibration modes.

CalibrationSucceeded

Getter method for the **CalibrationSucceeded** property. This property is the flag indicating if the calibration has succeeded (**TRUE**), that is whether the mean variation of grid points (distance between computed grid points and ideal grid points in world space) and the maximum variation of grid points are between given tolerances.

Closest

Find the daughter shape that is the closest to this shape. To retrieve the closest shape, use [EShape::ClosestShape](#).

DisableTypeFilter

Enables all shape types

Drag

Moves a handle to a new position and updates the position parameters of the shape.

DragLandmark

Moves the landmark to a new position.

Draw

Draws the world coordinate axis.

DrawCrossGrid

Draws a regular grid of crosses in world coordinates.

DrawCrossGridWithCurrentPen

Draws a regular grid of crosses in world coordinates.

DrawGrid

Draws the reconstructed grid to be used for grid calibration.

DrawGridWithCurrentPen

Draws the reconstructed grid to be used for grid calibration.

DrawLandmarks

Draws the landmarks to be used for landmark calibration.

DrawWithCurrentPen

Draws the world coordinate axis.

EmptyLandmarks

Resets the landmark specification sequence.

EnableTypeFilter

Enables the filter of the specified shape type

EWorldShape

Constructs a EWorldShape object.

GetLandmarkElement

Returns the landmark element corresponding to the given index.

HitLandmarks

Checks if the cursor is placed over a landmark point.

HitTest

Checks if there is a handle under the cursor.

operator=

Copies all the data from another EWorldShape object into the current EWorldShape object

RebuildGrid

Reconstructs the grid of points from the given dot centers, to compute the World coordinates of the points.

RemoveLandmark

Removes a landmark.

SensorToWorld

Performs coordinate transform for arbitrary points from Sensor space to World space.

SetCenterXY

Position of the origin point of the reference frame as projected onto the image, i.e. the Sensor position of the point at World coordinates **(0,0)**.

SetDistortion

Sets the optical distortion parameters

SetFieldSize

Sets the field of view size in physical units.

SetPan

Sets the horizontal and vertical panning factors for drawing operations.

SetPerspective

Sets the perspective effect coefficient, i.e. the inverse of the observation distance.

SetResolution

Sets the sensor resolution in pixels per unit in both directions.

SetSensor

Initializes the calibration object using all given parameters.

SetSensorSize

Sets the logical image size, i.e. the number of pixels horizontally and vertically.

SetSize

Sets the frame size.

SetupUnwarp

Prepares a lookup table for fast image unwarping.

SetZoom

Sets the horizontal and vertical zooming factors for drawing operations.

Unwarp

Unwarps a distorted image using the current calibration model.

WorldToSensor

Performs coordinate transform for arbitrary points from World space to Sensor space.

EWorldShape.AddLandmark

Adds a new pair of points coordinates (in Sensor and World spaces) to the set of landmarks used for calibration.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void AddLandmark(
    Euresys.Open_eVision_2_10.EPoint sensorPoint,
    Euresys.Open_eVision_2_10.EPoint worldPoint
)
```

Parameters

sensorPoint

Sensor point coordinates.

worldPoint

Corresponding World point coordinates.

Remarks

Landmark calibration is the process of computing the calibration parameters by means of a set of known points for which the coordinates are available in both World and Sensor spaces. Usually, such points are chosen as salient features on the part or target in view. They must be such that appropriate image processing techniques allow measuring their positions from the image (directly or indirectly by geometric constructions), while at the same time their coordinates in a reference frame are known.

EWorldShape.AddPoint

Adds a new point coordinates (in Sensor space) to the set of grid points used for calibration.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
void AddPoint(
    Euresys.Open_eVision_2_10.EPoint sensorPoint
)
```

Parameters

sensorPoint

Sensor point coordinates.

Remarks

Grid calibration is the process of computing the calibration parameters by means of a set of points known to lie on a rectangular grid. If the grid pitch is known and one of the points is chosen as the origin point, the points can be used as landmarks. By contrast with the landmark calibration functions, the World coordinates of the grid points need not be specified, nor do they have to be given in any specific order. The calibration algorithm is capable of sorting out the points to reconstruct the grid topology. Typically, this function is used in conjunction with blob analysis to extract the dot centers from a grid of dots. Anyway, any other scheme can be used. The grid of points need not be complete, i.e. some of the nodes may be missing, and the points need not completely fill a rectangular area. Landmark calibration is simply achieved by providing a series of point coordinates (in Sensor space only) and then calling the grid reconstruction function followed by the calibration function.

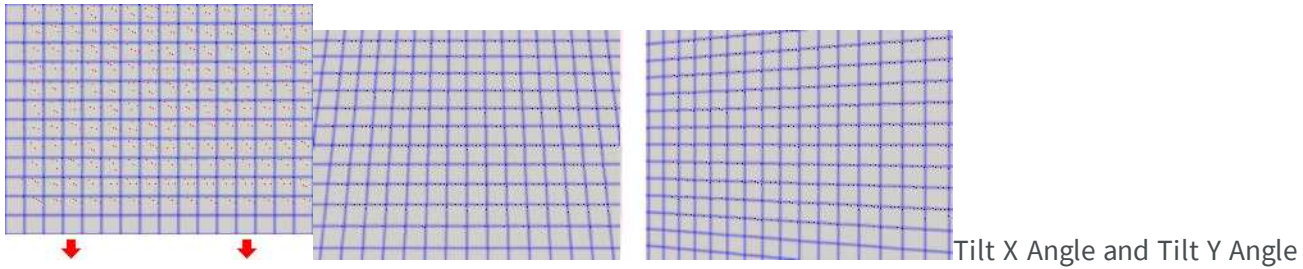
EWorldShape.Angle

Tilt X angle, that is the amplitude of the rotation applied around the X-axis of the sensor to bring the optical (Z) axis perpendicular to the field of view.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float Angle
{ get; set; }
```

Remarks



EWorldShape.AutoCalibrate

Returns the best calibration modes for the current calibration grid and calibrates the field of view accordingly.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint AutoCalibrate(
    bool testEmpiricalModes
)
```

Parameters

testEmpiricalModes

Boolean indicating whether empirical calibration modes ([Quadratic](#) and [Bilinear](#)) should be considered when determining the best calibration modes.

EWorldShape.AutoCalibrateDotGrid

Performs an automatic calibration based on a dot grid image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint AutoCalibrateDotGrid(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    float columnPitch,
    float rowPitch,
    bool testEmpiricalModes
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

columnPitch

Actual pitches of the grid, i.e. distances between vertical and horizontal rows of the grid.

rowPitch

Actual pitches of the grid, i.e. distances between vertical and horizontal rows of the grid.

testEmpiricalModes

Boolean indicating whether empirical calibration modes ([Quadratic](#) and [Bilinear](#)) should be considered when determining the best calibration modes. Default value is **FALSE**.

Remarks

Returns the best calibration mode for the current dot grid. The [EWorldShape::AutoCalibrateDotGrid](#) method will first do an automatic blob analysis in order to extract all dots (all blobs whose area is smaller than 5 pixels will be considered as noise and rejected). The dot gravity centers are used as the grid reference points. Then, the [EWorldShape::AutoCalibrateDotGrid](#) method will select and compute the best calibration mode by reducing the fitting error.

EWorldShape.AutoCalibrateLandmarks

Returns the best calibration modes for the current landmark set and calibrates the field of view accordingly.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
uint AutoCalibrateLandmarks(  
    bool testEmpiricalModes  
)
```

Parameters

testEmpiricalModes

Boolean indicating whether empirical calibration modes ([Quadratic](#) and [Bilinear](#)) should be considered when determining the best calibration modes. Default value is **FALSE**.

Remarks

Landmark calibration is the process of computing the calibration parameters by means of a set of known points for which the coordinates are available in both World and Sensor spaces. Usually, such points are chosen as salient features on the part or target in view. They must be such that appropriate image processing techniques allow measuring their positions from the image (directly or indirectly by geometric constructions), while at the same time their coordinates in a reference frame are known. The [EWorldShape::AutoCalibrateLandmarks](#) method is meant to be used with landmark calibration only. To calibrate automatically your field of view using a dot grid, use the [EWorldShape::AutoCalibrate](#) method instead.

EWorldShape.Calibrate

Performs a calibration according to the specified combination of calibration modes.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void Calibrate(  
    uint calibrationModes  
)
```

Parameters

calibrationModes

Calibration modes, as defined by a combination of values from [ECalibrationMode](#).

Remarks

Landmark calibration is the process of computing the calibration parameters by means of a set of known points for which the coordinates are available in both World and Sensor spaces. Usually, such points are chosen as salient features on the part or target in view. They must be such that appropriate image processing techniques allow measuring their positions from the image (directly or indirectly by geometric constructions), while at the same time their coordinates in a reference frame are known. In some cases, not all requested calibration modes are honored. After calibration, [EWorldShape::CalibrationModes](#) returns the actual combination of modes in effect.

EWorldShape.CalibrationModes

Current calibration mode, made from a combination of values.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint CalibrationModes
{ get; set; }
```

Remarks

The supported calibration modes can be set to [Raw](#), meaning that no calibration at all is performed (the World coordinates are pixel indices), or to the logical sum of other values from [ECalibrationMode](#).

EWorldShape.CalibrationSucceeded

Getter method for the **CalibrationSucceeded** property. This property is the flag indicating if the calibration has succeeded (**TRUE**), that is whether the mean variation of grid points (distance between computed grid points and ideal grid points in world space) and the maximum variation of grid points are between given tolerances.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool CalibrationSucceeded(
)
```

Remarks

The mean and maximum grid point variations are normalized using the pitch values. By default, tolerances are set to **0.05** (5 %) for the mean, and **0.1** (10 %) for the maximum grid point variation. You can get and set these tolerances using the [EWorldShape::GridPointsMeanVariationThreshold](#) and [EWorldShape::GridPointsMaxVariationThreshold](#) properties.

EWorldShape.Center

Position of the origin point of the reference frame as projected onto the image, i.e. the Sensor position of the point at World coordinates **(0,0)**.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.EPoint Center
{ get; set; }
```

EWorldShape.CenterX

Horizontal position of the origin point of the reference frame as projected onto the image, i.e. the Sensor abscissa of the point at World coordinates **(0,0)**.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float CenterX
{ get; }
```

EWorldShape.CenterY

Vertical position of the origin point of the reference frame as projected onto the image, i.e. the Sensor ordinate of the point at World coordinates **(0,0)**.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float CenterY
{ get; }
```

EWorldShape.Closest

Find the daughter shape that is the closest to this shape. To retrieve the closest shape, use [EShape::ClosestShape](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Closest(
)
```

EWorldShape.DisableTypeFilter

Enables all shape types

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void DisableTypeFilter(  
)
```

EWorldShape.DistortionStrength

Optical distortion strength, i.e. the ratio of the image diagonal length with and without optical distortion introduced by the lens.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float DistortionStrength  
{ get; }
```

EWorldShape.DistortionStrength2

Optical distortion strength of the second order.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
float DistortionStrength2
{ get; }
```

EWorldShape.Drag

Moves a handle to a new position and updates the position parameters of the shape.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Drag(
    int n32CursorX,
    int n32CursorY
)
```

Parameters

n32CursorX

Current cursor coordinates.

n32CursorY

Current cursor coordinates.

EWorldShape.DragLandmark

Moves the landmark to a new position.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DragLandmark(
    int n32CursorX,
    int n32CursorY
)
```

Parameters

n32CursorX

Current cursor coordinates.

n32CursorY

Current cursor coordinates.

EWorldShape.Draw

Draws the world coordinate axis.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingModes,
    bool daughters
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color,
    Euresys.Open_eVision_2_10.EDrawingMode drawingModes,
    bool daughters
)

void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingModes,
    bool daughters
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingModes

Indicates how the world coordinate axis must be displayed, as defined by [EDrawingMode](#).

daughters

Indicates whether the daughter shapes are to be displayed as well.

color

The color in which to draw the overlay.

EWorldShape.DrawCrossGrid

Draws a regular grid of crosses in world coordinates.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void DrawCrossGrid(  
    IntPtr graphicContext,  
    float minimumX,  
    float maximumX,  
    float minimumY,  
    float maximumY,  
    uint numberOfIntervalsX,  
    uint numberOfIntervalsY  
)  
  
void DrawCrossGrid(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.ERGBColor color,  
    float minimumX,  
    float maximumX,  
    float minimumY,  
    float maximumY,  
    uint numberOfIntervalsX,  
    uint numberOfIntervalsY  
)
```

```
void DrawCrossGrid(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    float minimumX,  
    float maximumX,  
    float minimumY,  
    float maximumY,  
    uint numberOfIntervalsX,  
    uint numberOfIntervalsY  
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

minimumX

Abscissa of the leftmost crosses, in world coordinates.

maximumX

Abscissa of the rightmost crosses, in world coordinates.

minimumY

Ordinate of the leftmost crosses, in world coordinates.

maximumY

Ordinate of the rightmost crosses, in world coordinates.

numberOfIntervalsX

Number of intervals between crosses along the horizontal direction.

numberOfIntervalsY

Number of intervals between crosses along the vertical direction.

color

The color in which to draw the overlay.

EWorldShape.DrawCrossGridWithCurrentPen

Draws a regular grid of crosses in world coordinates.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawCrossGridWithCurrentPen (
    IntPtr graphicContext,
    float minimumX,
    float maximumX,
    float minimumY,
    float maximumY,
    uint numberOfIntervalsX,
    uint numberOfIntervalsY
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

minimumX

Abscissa of the leftmost crosses, in world coordinates.

maximumX

Abscissa of the rightmost crosses, in world coordinates.

minimumY

Ordinate of the leftmost crosses, in world coordinates.

maximumY

Ordinate of the rightmost crosses, in world coordinates.

numberOfIntervalsX

Number of intervals between crosses along the horizontal direction.

numberOfIntervalsY

Number of intervals between crosses along the vertical direction.

EWorldShape.DrawGrid

Draws the reconstructed grid to be used for grid calibration.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawGrid(
    IntPtr graphicContext
)

void DrawGrid(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.ERGBColor color
)

void DrawGrid(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

color

The color in which to draw the overlay.

EWorldShape.DrawGridWithCurrentPen

Draws the reconstructed grid to be used for grid calibration.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawGridWithCurrentPen(
    IntPtr graphicContext
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

EWorldShape.DrawLandmarks

Draws the landmarks to be used for landmark calibration.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawLandmarks (
    IntPtr graphicContext
)
void DrawLandmarks (
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

EWorldShape.DrawWithCurrentPen

Draws the world coordinate axis.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void DrawWithCurrentPen (
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EDrawingMode drawingModes,
    bool daughters
)
```

Parameters

graphicContext

Handle of the device context on which to draw.

drawingModes

Indicates how the world coordinate axis must be displayed, as defined by [EDrawingMode](#).

daughters

Indicates whether the daughter shapes are to be displayed as well.

EWorldShape.EmptyLandmarks

Resets the landmark specification sequence.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void EmptyLandmarks (  
)
```

Remarks

Landmark calibration is the process of computing the calibration parameters by means of a set of known points for which the coordinates are available in both World and Sensor spaces. Usually, such points are chosen as salient features on the part or target in view. They must be such that appropriate image processing techniques allow measuring their positions from the image (directly or indirectly by geometric constructions), while at the same time their coordinates in a reference frame are known.

EWorldShape.EnableTypeFilter

Enables the filter of the specified shape type

Namespace: Euresys.Open_eVision_2_10


```
[C#]
void EnableTypeFilter (
    uint un32Types
)
```

Parameters

un32Types

The type of the shape to filter from [EShapeType](#).

EWorldShape.EWorldShape

Constructs a EWorldShape object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EWorldShape (
    Euresys.Open_eVision_2_10.EWorldShape other
)
void EWorldShape (
)
```

Parameters

other

Another EWorldShape object to be copied in the new EWorldShape object.

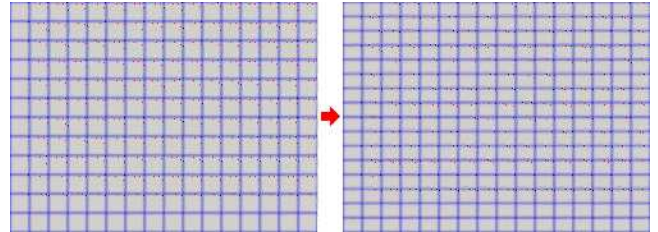
EWorldShape.FieldHeight

Field-of-view height, in physical units.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float FieldHeight
{ get; }
```

Remarks



Field size not matching the sensor size results in non-square pixels.

Pixels having non-square aspect ratio Beware there is a restriction pertaining to the allowed image anisotropy. The resulting pixels aspect ratio (**XResolution/YResolution**) should be in the range **[-4/3, -3/4]** (or **[3/4, 4/3]**), otherwise the calibration process could fail.

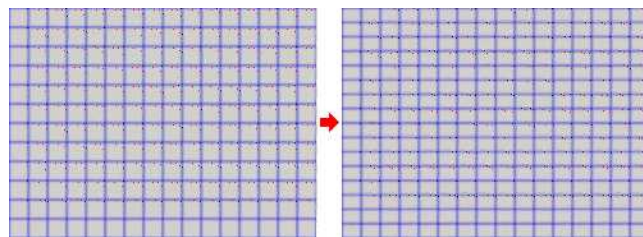
EWorldShape.FieldWidth

Field-of-view width, in physical units.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float FieldWidth
{ get; }
```

Remarks



Field size not matching the sensor size results in non-square pixels.

Pixels having non-square aspect ratio Beware there is a restriction pertaining to the allowed image anisotropy. The resulting pixels aspect ratio (**XResolution/YResolution**) should be in the range **[-4/3, -3/4]** (or **[3/4, 4/3]**), otherwise the calibration process could fail.

EWorldShape.GetLandmarkElement

Returns the landmark element corresponding to the given index.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
Euresys.Open_eVision_2_10.ELandmark GetLandmarkElement(
    uint i
)
Euresys.Open_eVision_2_10.ELandmark GetLandmarkElement(
    uint i
)
```

Parameters

i

Landmark index.

EWorldShape.GridPointsMaxVariation

Maximum variation of grid points (maximum distance between computed grid points and ideal grid points in world space), normalized using the pitch values, after a call to [EWorldShape::CalibrationSucceeded](#)

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float GridPointsMaxVariation  
  
{ get; }
```

Remarks

The maximum grid point variation is normalized using the pitch values.

EWorldShape.GridPointsMaxVariationThreshold

Grid points maximum variation threshold, that is the value above which the maximum variation of grid points (maximum distance between computed grid points and ideal grid points in world space) is considered too high, and thus marks the calibration as a failure when using [EWorldShape::CalibrationSucceeded](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float GridPointsMaxVariationThreshold  
  
{ get; set; }
```

Remarks

The maximum grid point variation is normalized using the pitch values.

EWorldShape.GridPointsMeanVariation

Mean variation of grid points (mean distance between computed grid points and ideal grid points in world space), normalized using the pitch values, after a call to [EWorldShape::CalibrationSucceeded](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float GridPointsMeanVariation
{ get; }
```

Remarks

The mean grid point variation is normalized using the pitch values.

EWorldShape.GridPointsMeanVariationThreshold

Grid points mean variation threshold, that is the value above which the mean variation of grid points (mean distance between computed grid points and ideal grid points in world space) is considered too high, and thus marks the calibration as a failure when using [EWorldShape::CalibrationSucceeded](#).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float GridPointsMeanVariationThreshold
{ get; set; }
```

Remarks

The mean grid point variation is normalized using the pitch values.

EWorldShape.HitLandmark

Returns the landmark selected by [EWorldShape::HitLandmarks](#) or ~0 if no landmark is selected.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint HitLandmark
{ get; }
```

EWorldShape.HitLandmarks

Checks if the cursor is placed over a landmark point.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void HitLandmarks (
)
```

Remarks

Landmark calibration is the process of computing the calibration parameters by means of a set of known points for which the coordinates are available in both World and Sensor spaces. Usually, such points are chosen as salient features on the part or target in view. They must be such that appropriate image processing techniques allow measuring their positions from the image (directly or indirectly by geometric constructions), while at the same time their coordinates in a reference frame are known.

EWorldShape.HitTest

Checks if there is a handle under the cursor.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool HitTest(
    bool bDaughters
)
```

Parameters

bDaughters

Indicates if the check must be done in the whole hierarchy or just this object.

EWorldShape.NumLandmarkElements

Returns the number of landmark elements.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint NumLandmarkElements
{ get; }
```

EWorldShape.operator=

Copies all the data from another EWorldShape object into the current EWorldShape object

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EWorldShape operator=(  
    Euresys.Open_eVision_2_10.EWorldShape other  
)
```

Parameters

other

EWorldShape object to be copied

EWorldShape.PanX

Current horizontal panning factor for drawing operations.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
override float PanX  
  
    { get; }
```

EWorldShape.PanY

Current vertical panning factor for drawing operations.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
override float PanY  
  
    { get; }
```


EWorldShape.PerspectiveStrength

Perspective effect coefficient, that is the inverse of the observation distance.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float PerspectiveStrength  
  
    { get; }
```

Remarks

The larger this parameter, the more perceivable the perspective distortion will be. A **NULL** value corresponds to a telecentric lens.

EWorldShape.Ratio

XResolution/YResolution ratio.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Ratio  
  
    { get; set; }
```

Remarks

If **Ratio** equals **-1** (or **1**), pixels are square. Beware there is a restriction pertaining to the allowed image anisotropy. The resulting pixels aspect ratio (**XResolution/YResolution**) should be in the range **[-4/3, -3/4]** (or **[3/4, 4/3]**), otherwise the calibration process could fail.

EWorldShape.RebuildGrid

Reconstructs the grid of points from the given dot centers, to compute the World coordinates of the points.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
uint RebuildGrid(
    float colPitch,
    float rowPitch,
    uint centerIndex,
    bool direct
)

uint RebuildGrid(
    float colPitch,
    float rowPitch,
    Euresys.Open_eVision_2_10.EPoint worldCenter,
    uint centerIndex,
    bool direct
)
```

Parameters

colPitch

Actual pitches of the grid, that are distances between vertical and horizontal rows of the grid.

rowPitch

Actual pitches of the grid, that are distances between vertical and horizontal rows of the grid.

centerIndex

Index of the grid point chosen as coordinate origin point. By default, the most central grid point.

direct

TRUE if the world reference frame points upwards.

worldCenter

World coordinates of the starting grid point.

Remarks

This member function also returns the number of grid points that were connected. This prepares the calibration using landmarks (for use by member [EWorldShape::Calibrate](#)). Landmark calibration is the process of computing the calibration parameters by means of a set of known points for which the coordinates are available in both World and Sensor spaces. Usually, such points are chosen as salient features on the part or target in view. They must be such that appropriate image processing techniques allow measuring their positions from the image (directly or indirectly by geometric constructions), while at the same time their coordinates in a reference frame are known. See also Dot-Grid-Based Calibration for the grid construction algorithm.

EWorldShape.RemoveLandmark

Removes a landmark.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void RemoveLandmark(  
    uint index  
)
```

Parameters

index

Index of the landmark to be removed.

EWorldShape.Scale

The scale of the frame.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float Scale
```

```
{ get; set; }
```

EWorldShape.SensorHeight

Logical image height, that is the number of pixels vertically.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
int SensorHeight
```

```
{ get; }
```

EWorldShape.SensorToWorld

Performs coordinate transform for arbitrary points from Sensor space to World space.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EPoint SensorToWorld(  
    Euresys.Open_eVision_2_10.EPoint sensorPoint  
)
```

Parameters

sensorPoint

Sensor point.

EWorldShape.SensorWidth

Logical image width, that is the number of pixels horizontally.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
int SensorWidth
{ get; }
```

EWorldShape.SetCenterXY

Position of the origin point of the reference frame as projected onto the image, i.e. the Sensor position of the point at World coordinates **(0,0)**.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetCenterXY(
    float centerX,
    float centerY
)
```

Parameters

centerX

Horizontal position (abscissa)

centerY

Vertical position (ordinate)

EWorldShape.SetDistortion

Sets the optical distortion parameters

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void SetDistortion(  
    float distortionStrength,  
    float distortionStrength2  
)
```

Parameters

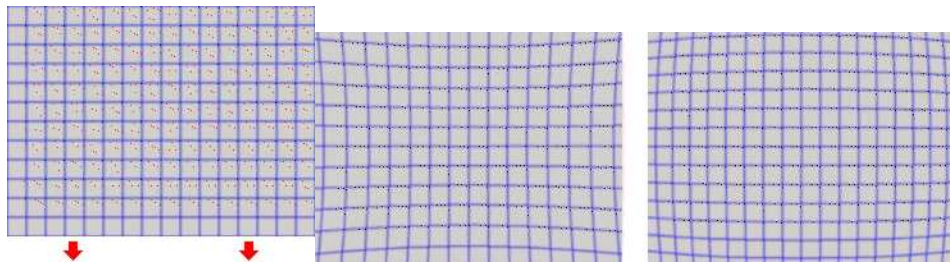
distortionStrength

Optical distortion strength, i.e. the ratio of the image diagonal length with and without optical distortion introduced by the lens.

distortionStrength2

Optical distortion strength of the second order.

Remarks



Positive distortion and negative distortion

EWorldShape.SetFieldSize

Sets the field of view size in physical units.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetFieldSize(
    float width,
    float height
)
```

Parameters

width

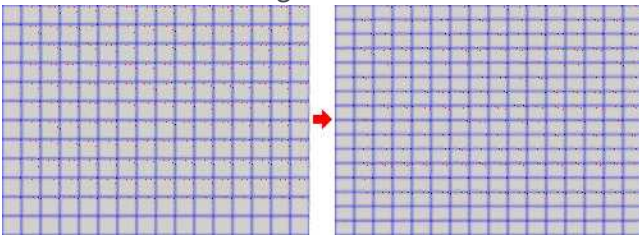
Full image physical width, in length units.

height

Full image physical height, in length units. If not specified, same as physical width.

Remarks

Field size not matching the sensor size results in non-square pixels. By default, the pixels are square.



Pixels having non-square aspect ratio Beware there is a restriction pertaining to the allowed image anisotropy. The resulting pixels aspect ratio (**XResolution/YResolution**) should be in the range **[-4/3, -3/4]** (or **[3/4, 4/3]**), otherwise the calibration process could fail.

EWorldShape.SetPan

Sets the horizontal and vertical panning factors for drawing operations.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetPan(
    float panX,
    float panY
)
```

Parameters

panX

Horizontal panning factor. By default, no panning occurs.

panY

Vertical panning factor. By default, no panning occurs.

Remarks

All objects attached to an [EWorldShape](#) object inherit of the same panning factor.

EWorldShape.SetPerspective

Sets the perspective effect coefficient, i.e. the inverse of the observation distance.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetPerspective(
    float tiltXAngle,
    float tiltYAngle,
    float perspectiveStrength
)
```

Parameters

tiltXAngle

Tilt angles, i.e. the amplitudes of the rotations applied around the X and Y axis of the sensor to bring the optical (Z) axis perpendicular to the field of view.

tiltYAngle

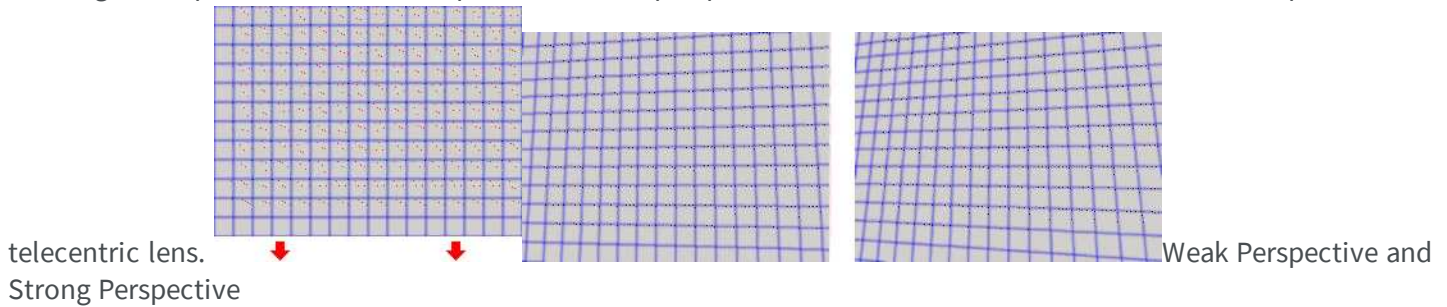
Tilt angles, i.e. the amplitudes of the rotations applied around the X and Y axis of the sensor to bring the optical (Z) axis perpendicular to the field of view.

perspectiveStrength

Perspective effect coefficient.

Remarks

The larger this parameter, the more perceivable the perspective distortion will be. A **NULL** value corresponds to a



EWorldShape.SetResolution

Sets the sensor resolution in pixels per unit in both directions.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void SetResolution(  
    float resolutionX,  
    float resolutionY  
)
```

Parameters

resolutionX

Horizontal resolution in pixels per units

resolutionY

Vertical resolution in pixels per units. If not specified, same as horizontal resolution.

Remarks

By default, the pixels are square.

EWorldShape.SetSensor

Initializes the calibration object using all given parameters.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetSensor(
    int sensorWidth,
    int sensorHeight,
    float fieldWidth,
    float fieldHeight,
    float centerX,
    float centerY,
    float angle,
    float tiltXAngle,
    float tiltYAngle,
    float perspectiveStrength,
    float distortionStrength,
    float distortionStrength2,
    float opticalCenterX,
    float opticalCenterY,
    uint calibrationModes
)
```

Parameters

sensorWidth

Logical size of the field of view, i.e. image size, in pixels.

sensorHeight

Logical size of the field of view, i.e. image size, in pixels.

fieldWidth

Physical size of the field of view. By default (argument omitted), the pixels are square.

fieldHeight

Physical size of the field of view. By default (argument omitted), the pixels are square.

centerX

Position of the "intersection" between the optical axis and the field of view in the image. By default, if the calibration modes contain [Raw](#), it is set to 0. Otherwise, it is set to the image center.

centerY

Position of the "intersection" between the optical axis and the field of view in the image. By default, if the calibration modes contain [Raw](#), it is set to 0 (or to the bottommost pixel index if the calibration modes also contain [Inverse](#)). Otherwise, it is set to the image center.

angle

Skew angle, i.e. angle formed by the axis of reference and the image edges. By default (argument omitted), no skewing effect is assumed.

tiltXAngle

Rotation angles on the X axis to bring the optical axis perpendicular to the image plane. By default (argument omitted), no perspective effect is assumed.

tiltYAngle

Rotation angles on the Y axis to bring the optical axis perpendicular to the image plane. By default (argument omitted), no perspective effect is assumed.

perspectiveStrength

Relative importance of the perspective effect. By default, no perspective effect is assumed, as if the lens was telecentric.

distortionStrength

Relative importance of the lens radial distortion. Positive for barrel, negative for cushion. By default (argument omitted), no optical distortion is assumed.

distortionStrength2

Relative importance of the lens radial distortion of the second order. By default (argument omitted), no optical distortion of second order is assumed.

opticalCenterX

X Position of the "intersection" between the optical axis and the field of view in the image. By default (argument omitted) the image center.

opticalCenterY

Y Position of the "intersection" between the optical axis and the field of view in the image. By default (argument omitted) the image center.

calibrationModes

Desired calibration mode effects to be combined, as defined by [ECalibrationMode](#). By default (argument omitted), the simplest model compatible with the given parameters is chosen.

Remarks

The function automatically selects the appropriate calibration model by checking the parameters. The use of a more complex calibration mode can be enforced by means of parameter [EWorldShape::CalibrationModes](#), not a simpler one.

EWorldShape.SetSensorSize

Sets the logical image size, i.e. the number of pixels horizontally and vertically.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void SetSensorSize(  
    int width,  
    int height  
)
```

Parameters

width

Full image logical sizes, in pixels.

height

Full image logical sizes, in pixels.

EWorldShape.SetSize

Sets the frame size.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void SetSize(  
    float sizeX,  
    float sizeY  
)
```

Parameters

sizeX

Frame X-axis length. The default value is **100**.

sizeY

Frame Y-axis length. By default, both axes have the same length.

Remarks

By default, both frame axis value are set to **100**, which means 100 pixels when the field of view is not calibrated and 100 "units" in case of a calibrated field of view.

EWorldShape.SetupUnwarp

Prepares a lookup table for fast image unwarping.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetupUnwarp(
    Euresys.Open_eVision_2_10.EUnwarpingLut lookupTable,
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    bool interpolate
)

void SetupUnwarp(
    Euresys.Open_eVision_2_10.EUnwarpingLut lookupTable,
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    bool interpolate
)
```

Parameters

lookupTable

Pointer to the lookup table.

sourceImage

Pointer to the source image/ROI.

interpolate

Interpolation mode. Default value is **FALSE**.

Remarks

The function should be called each time the system is re-calibrated (after the optical setup has been changed, for instance). A sample source image has to be supplied to [EWorldShape::SetupUnwarp](#), and its row pitch is recorded in order to speedup the unwarping process. This implies that the following calls to [EWorldShape::Unwarp](#) are not allowed to use images with row pitches different from the source image initially supplied to [EWorldShape::SetupUnwarp](#).

EWorldShape.SetZoom

Sets the horizontal and vertical zooming factors for drawing operations.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void SetZoom(
    float zoomX,
    float zoomY
)
```

Parameters

zoomX

Horizontal zooming factor. By default, true scale is used.

zoomY

Vertical zooming factor. If set to **0**, the default value, the horizontal zooming factor is used instead, so as to provide isotropic zooming.

Remarks

All objects attached to an [EWorldShape](#) inherit of the same zooming factor.

EWorldShape.TiltXAngle

Tilt X angle, that is the amplitude of the rotation applied around the X-axis of the sensor to bring the optical (Z) axis perpendicular to the field of view.

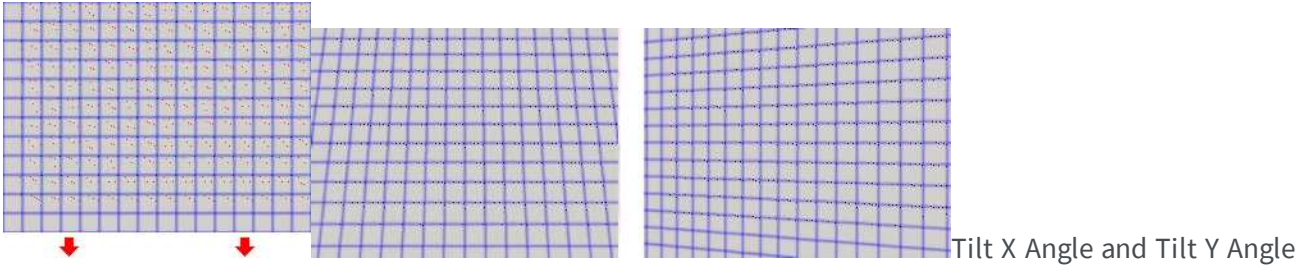
Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float TiltXAngle
```

```
{ get; }
```

Remarks



EWorldShape.TiltYAngle

Tilt Y angle, that is the amplitude of the rotation applied around the Y-axis of the sensor to bring the optical (Z) axis perpendicular to the field of view.

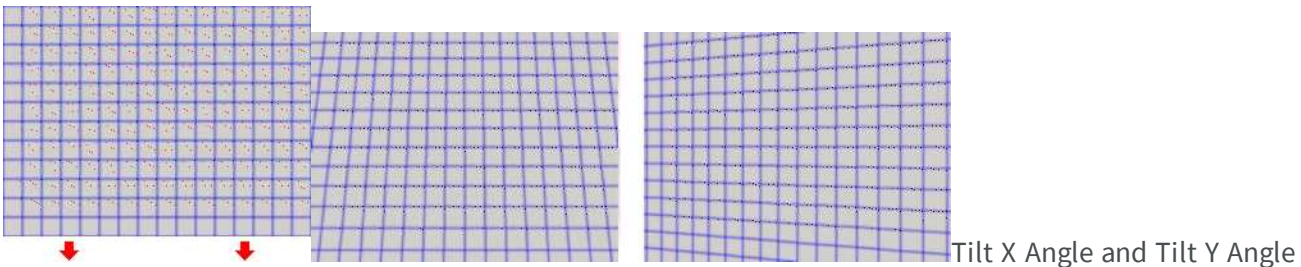
Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float TiltYAngle
```

```
{ get; }
```

Remarks



EWorldShape.Type

Shape type.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
override Euresys.Open_eVision_2_10.EShapeType Type
{ get; }
```

EWorldShape.Unwarp

Unwarps a distorted image using the current calibration model.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void Unwarp(
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,
    bool interpolate
)

void Unwarp(
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,
    bool interpolate
)
```



```
void Unwarp(  
    Euresys.Open_eVision_2_10.EUnwarpingLut lookupTable,  
    Euresys.Open_eVision_2_10.EROIBW8 sourceImage,  
    Euresys.Open_eVision_2_10.EROIBW8 destinationImage,  
    bool interpolate  
)  
  
void Unwarp(  
    Euresys.Open_eVision_2_10.EUnwarpingLut lookupTable,  
    Euresys.Open_eVision_2_10.EROIC24 sourceImage,  
    Euresys.Open_eVision_2_10.EROIC24 destinationImage,  
    bool interpolate  
)
```

Parameters

sourceImage

Pointer to the source image/ROI.

destinationImage

Pointer to the destination unwarped image.

interpolate

Interpolation mode. Default value is **FALSE**.

lookupTable

Pointer to the lookup table.

Remarks

Using a pre-computed lookup table allows speeding up the unwarping process. The lookup table is initialized by means of the [EWorldShape::SetupUnwarp](#) function.

EWorldShape.WorldToSensor

Performs coordinate transform for arbitrary points from World space to Sensor space.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
Euresys.Open_eVision_2_10.EPoint WorldToSensor(  
    Euresys.Open_eVision_2_10.EPoint worldPoint  
)
```

Parameters

worldPoint
World point.

EWorldShape.XResolution

Horizontal sensor resolution, in pixels per unit.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float XResolution  
    { get; }
```

EWorldShape.YResolution

Vertical sensor resolution, in pixels per unit.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float YResolution  
    { get; }
```

EWorldShape.ZoomX

Current horizontal zooming factor for drawing operations.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
override float ZoomX  
  
    { get; }
```

EWorldShape.ZoomY

Current vertical zooming factor for drawing operations.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
override float ZoomY  
  
    { get; }
```

4.194. EZMap Class

Represents a generic ZMap type interface.

Derived Class(es): [EZMap8](#) [EZMap16](#) [EZMap32f](#)

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

Height

Access ZMap Height.

MapToWorldMatrix

Returns an [E3DTransformMatrix](#) that transforms positions from the [EZMap](#) space to the world space. This transformation is composed of rotation and translation only, so it is a rigid transformation and preserves distances.

RowPitch

Returns the buffer row pitch.

Type

Pixel accessor type.

Width

Access ZMap Width.

WorldShape

Returns the [EWorldShape](#) for conversion between 2D image and 2D world space coordinates. This [EWorldShape](#) can be used by [EasyGauge](#) to do measurements on a [EZMap](#) in real space coordinates (e.g mm).

WorldToMapMatrix

Returns an [E3DTransformMatrix](#) that transforms positions from the world space to the [EZMap](#) space. This transformation is composed of rotation and translation only, so it is a rigid transformation and it preserves distances.

XResolution

Resolution of the [EZMap](#) along the X axis
The resolution is the number of metric units per pixel.

YResolution

Resolution of the [EZMap](#) along the Y axis
The resolution is the number of metric units per pixel.

ZResolution

M Resolution of the [EZMap](#) along the Z axis
The resolution is the number of grey values per pixel.

e

thods

AddMetadata

Adds a metadata key (name) and value.
If the metadata key already exists, its value will be overwritten.

Clear

Clears the ZMap: replaces all pixels with the undefined value.

Create

Factory method to allocates and reads a ZMap from a file. Depending of the serialized EZMap type, it returns the corresponding [EZMap8](#), [EZMap16](#) or [EZMap32f](#) object. The allocated EZMap must be released after use.

Draw

Draws a [EZMap](#) in a device context.

DrawImage

Displays the internal image buffer

GetBufferPtr

Retrieves the pointer to the internal pixel buffer.

GetCheckedBufferPtr

Retrieves the pointer to the pixel buffer.

GetMetadata

Returns the string value of the given metadata.
Throws an exception if it does not exist.

GetResolution

Gets the resolution of the [EZMap](#) along the X, Y and Z axis.
On the Z axis, the resolution is the number of metric units per grey value.
On the X and Y axis, the resolution is the number of metric units per pixel.

GetSizeInWorld

Returns the dimensions of the [EZMap](#) in real world space (e.g metric unit).

GetWorldPositionFromPixelPosition

Returns the 3D world position corresponding to a [EZMap](#) pixel position. The world position is in the original point cloud space. The pixel space origin is at the upper left corner of the image and ranges to (width-1, height-1). The transformation to the world position is calculated using the center of the pixel. The pixel value at position (u,v) must not be undefined, otherwise an exception will be thrown.

GetZMapPositionFromPixelPosition

Returns the corresponding [EZMap](#) 3D position of a ZMap pixel. The ZMap position is in the original point ZMap space. The pixel position origin is the upper left corner of the image and ranges to (width-1, height-1). The center of the pixel is used for the transformation to the ZMap position. Pixel at position (u,v) must be defined, otherwise an exception will be thrown.

ImageToWorld

Transforms a floating point (sub)pixel image position to a 3D world position.
The image space origin is at the upper left corner of the image and the X/Y unit size is one pixel.
The pixel Z value is in grey scale values (its range depends on the ZMap type).

ImageToZMap

Converts a 2D image (sub)pixel coordinate to the [EZMap](#) space. (u,v) is the pixel position (with its origin in the upper left corner of the image).
(x,y) is the corresponding ZMap position (which has the same scale as the world space).
All values are expressed in floating point numbers.

IsVoid

Tests if the [EZMap](#) object size is zero.

Load

Restores the [EZMap](#) stored in the given Open eVision file.

LoadImage

Restores the [EZMap](#) image stored in the given image file.

LoadImageAndMetadata

Loads image format and Metadata in JSON format.

LoadMetadata

Loads Metadata in JSON format.

Save

Saves the [EZMap](#) object to the given Open eVision file.

SaveImage

Saves the [EZMap](#) image to the given image file.

SaveImageAndMetadata

Saves image format and Metadata JSON format.

SaveMetadata

Saves Metadata in JSON format.

Serialize

Serializes the [EZMap](#) object with all its attributes.

SerializeImage

Serializes the image associated to [EZMap](#).

SetBufferPtr

Sets the pointer to an externally allocated image buffer.

SetResolution

Sets the resolution of the [EZMap](#) along the X, Y and Z axis.
On the Z axis, the resolution is the number of metric units per grey value.
On the X and Y axis, the resolution is the number of metric units per pixel.

SetSize

Sets the width and height of the [EZMap](#).

WorldToImage

Transforms a 3D world position to a floating point (sub)pixel image position.
The image space origin is at the upper left corner of the image and the X/Y unit size is one pixel.
The Z value is given in grey scale value.
Returns TRUE if the pixel position is inside the image limits and the Z value is positive.
The parameter pixelPt is filled even if the point is outside the image.

WorldToZMap

Transforms a 3D world position to a 3D **EZMap** position.
The ZMap space origin is at the lower left corner of the image.
The scales of the world space and ZMap space are the same.

ZMapToImage

Converts a 2D coordinate in the **EZMap** space to image (sub)pixel space.
(x,y) is the ZMap position (which has the same scale as the world space).
(u,v) is the corresponding pixel position (with its origin in the upper left corner of the image).
All values are expressed in floating point numbers.
Returns TRUE if the pixel position is inside the image limits.

ZMapToWorld

Transforms a 3D **EZMap** position to a 3D world space position.
The ZMap space origin is at the lower left corner of the image.

Map.AddMetadata

Adds a metadata key (name) and value.
If the metadata key already exists, its value will be overwritten.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
void AddMetadata (  
    string Key,  
    string value  
)
```

Parameters

Key

The name of the metadata. Names are unique.

value

The value for the given metadata.

EZMap.Clear

Clears the ZMap: replaces all pixels with the undefined value.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
void Clear(  
    )
```

EZMap.Create

Factory method to allocates and reads a ZMap from a file. Depending of the serialized EZMap type, it returns the corresponding [EZMap8](#), [EZMap16](#) or [EZMap32f](#) object. The allocated EZMap must be released after use.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
Euresys.Open_eVision_2_10.Easy3D.EZMap Create(  
    string path  
    )
```

Parameters

path

Full path to the file.

EZMap.Draw

Draws a [EZMap](#) in a device context.

Namespace: Euresys.Open_eVision_2_10.Easy3D

[C#]

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

```

Parameters

graphicContext

Handle to the device context of the destination window.

zoomX

Magnification factor for zooming in or out in the horizontal direction. By default, the ZMap is displayed in 1:1 scale.

zoomY

Magnification factor for zooming in or out in the vertical direction. Setting a **0** value (which is the default) will result in isotropic scaling (i.e. equal horizontal and vertical factors).

panX

Pan offset (in pixels) in the horizontal direction. By default, no panning is applied.

panY

Pan offset (in pixels) in the vertical direction. By default, no panning is applied.

colorUndefinedPixel

An optional parameter to choose the drawing color of undefined pixels.

c24Vector

When supplied, this parameter allows using a LUT that maps from Depth to C24 when drawing (false colors).

bw8Vector

When supplied, this parameter allows using a LUT that maps from Depth to BW8 when drawing.

Remarks

A ZMap can be drawn (its pixels rendered) using a device context.

The horizontal and vertical zooming factors can be different but must be in the **1/16..16** range.

(MFC users can use the **CDC::GetSafeHdc()** method to obtain a suitable device context handle from a **CDC** instance.)

EZMap.DrawImage

Displays the internal image buffer

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void DrawImage (
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)
```

```
void DrawImage(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void DrawImage(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void DrawImage(  
    IntPtr graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void DrawImage(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void DrawImage(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EBw8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

Parameters

graphicContext

Handle to the device context of the destination window.

zoomX

Magnification factor for zooming in or out in the horizontal direction. By default, the image is displayed in 1:1 scale.

zoomY

Magnification factor for zooming in or out in the vertical direction. Setting a **0** value (which is the default) will result in isotropic scaling (i.e. equal horizontal and vertical factors).

panX

Pan offset (in pixels) in the horizontal direction. By default, no panning is applied.

panY

Pan offset (in pixels) in the vertical direction. By default, no panning is applied.

colorUndefinedPixel

An optional parameter to choose the drawing color of undefined pixels.

c24Vector

When supplied, this parameter allows using a LUT that maps from Depth to C24 when drawing (false colors).

bw8Vector

When supplied, this parameter allows using a LUT that maps from Depth to BW8 when drawing.

Remarks

An image can be drawn (its pixels rendered) using a device context.

The horizontal and vertical zooming factors can be different but must be in the **1/16..16** range.

(MFC users can use the **CDC::GetSafeHdc()** method to obtain a suitable device context handle from a **CDC** instance.)

EZMap.GetBufferPtr

Retrieves the pointer to the internal pixel buffer.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
IntPtr GetBufferPtr (
)
IntPtr GetBufferPtr (
    int x,
    int y
)
IntPtr GetBufferPtr (
)
IntPtr GetBufferPtr (
    int x,
    int y
)
```

Parameters

x

Column of the pixel which we want the address.

y

Row of the pixel which we want the address.

Remarks

This function does not check the value of the parameters.
Use carefully.

EZMap.GetCheckedBufferPtr

Retrieves the pointer to the pixel buffer.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
IntPtr GetCheckedBufferPtr (
    int x,
    int y
)

IntPtr GetCheckedBufferPtr (
    int x,
    int y
)
```

Parameters

x

Column of the pixel of which we want the address.

y

Row of the pixel of which we want the address.

EZMap.GetMetadata

Returns the string value of the given metadata.
Throws an exception if it does not exist.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
string GetMetadata (
    string Key
)
```

Parameters

Key

The name of an existing metadata.

EZMap.GetResolution

Gets the resolution of the [EZMap](#) along the X, Y and Z axis.
On the Z axis, the resolution is the number of metric units per grey value.
On the X and Y axis, the resolution is the number of metric units per pixel.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DPoint GetResolution(
)
void GetResolution(
    ref float sx,
    ref float sy,
    ref float sz
)
```

Parameters

sx

Contains the resolution along the X axis.

sy

Contains the resolution along the Y axis.

sz

Contains the resolution along the Z axis.

EZMap.GetSizeInWorld

Returns the dimensions of the [EZMap](#) in real world space (e.g metric unit).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void GetSizeInWorld(
    ref float worldWidth,
    ref float worldHeight
)
```

Parameters

worldWidth

Contains the size of the ZMap along the X axis (column).

worldHeight

Contains the size of the ZMap along the Y axis (row).

EZMap.GetWorldPositionFromPixelPosition

Returns the 3D world position corresponding to a [EZMap](#) pixel position. The world position is in the original point cloud space. The pixel space origin is at the upper left corner of the image and ranges to (width-1, height-1). The transformation to the world position is calculated using the center of the pixel. The pixel value at position (u,v) must not be undefined, otherwise an exception will be thrown.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DPoint GetWorldPositionFromPixelPosition(
    int u,
    int v
)
```

Parameters

u

Column of the pixel (bounds: [0,width]).

v

Row of the pixel (bounds: [0,height]).

EZMap.GetZMapPositionFromPixelPosition

Returns the corresponding [EZMap](#) 3D position of a ZMap pixel. The ZMap position is in the original point ZMap space. The pixel position origin is the upper left corner of the image and ranges to (width-1, height-1). The center of the pixel is used for the transformation to the ZMap position. Pixel at position (u,v) must be defined, otherwise an exception will be thrown.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DPoint GetZMapPositionFromPixelPosition(
    int u,
    int v
)
```

Parameters

u

Column of the pixel (bounds: [0,width]).

v

Row of the pixel (bounds: [0,height]).

EZMap.Height

Access ZMap Height.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
abstract int Height
{ get; set; }
```

EZMap.ImageToWorld

Transforms a floating point (sub)pixel image position to a 3D world position.
The image space origin is at the upper left corner of the image and the X/Y unit size is one pixel.
The pixel Z value is in grey scale values (its range depends on the ZMap type).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ImageToWorld(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint pixelPt,
    ref Euresys.Open_eVision_2_10.Easy3D.E3DPoint worldPt
)
```

Parameters

pixelPt

Position in the image space.

worldPt

Position in the 3D world space.

EZMap.ImageToZMap

Converts a 2D image (sub)pixel coordinate to the [EZMap](#) space.
(u,v) is the pixel position (with its origin in the upper left corner of the image).
(x,y) is the corresponding ZMap position (which has the same scale as the world space).
All values are expressed in floating point numbers.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
void ImageToZMap(  
    float u,  
    float v,  
    ref float x,  
    ref float y  
)
```

Parameters

u

X Coordinate of the pixel as a floating point value.

v

Y Coordinate of the pixel as a floating point value.

x

Position along horizontal axis in the ZMap space.

y

Position along vertical axis in the ZMap space.

EZMap.IsVoid

Tests if the [EZMap](#) object size is zero.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
bool IsVoid(  
)
```

Remarks

Returns **TRUE** if the ZMap size is zero.

EZMap.Load

Restores the [EZMap](#) stored in the given Open eVision file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Load(
    string path
)
```

Parameters

path
Full path to the file.

Remarks

When loading, the ZMap is resized if needed. This function restores all the ZMap attributes.

EZMap.LoadImage

Restores the [EZMap](#) image stored in the given image file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void LoadImage(
    string path
)
```

Parameters

path
Full path to the file.

Remarks

When loading, the ZMap is resized if needed. This function does not restore the ZMap attributes, only the image associated with the [EZMap](#) is updated.

EZMap.LoadImageAndMetadata

Loads image format and Metadata in JSON format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void LoadImageAndMetadata (
    string pathImage,
    string pathMetadata
)
```

Parameters

pathImage

Full path to the file.

pathMetadata

Full path to the file.

EZMap.LoadMetadata

Loads Metadata in JSON format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
void LoadMetadata(  
    string path  
)
```

Parameters

path

Full path to the file.

EZMap.MapToWorldMatrix

Returns an [E3DTransformMatrix](#) that transforms positions from the [EZMap](#) space to the world space. This transformation is composed of rotation and translation only, so it is a rigid transformation and preserves distances.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
abstract Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix MapToWorldMatrix  
    { get; }
```

EZMap.RowPitch

Returns the buffer row pitch.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
abstract int RowPitch  
    { get; }
```

EZMap.Save

Saves the [EZMap](#) object to the given Open eVision file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Save(
    string path
)
```

Parameters

path

The full path to the destination file.

Remarks

This format save the [EZMap](#) in a Open eVision file. This function stores all the ZMap attributes.

EZMap.SaveImage

Saves the [EZMap](#) image to the given image file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SaveImage(
    string path,
    Euresys.Open_eVision_2_10.EImageFileType type
)
```

Parameters

path

The full path to the destination file.

type

File format, as defined by [EImageFileType](#). If not specified, the file format is determined from the file extension.

Remarks

This format save the image associated to [EZMap](#) in a standard image file and thus does not store ZMap attributes.

EZMap.SaveImageAndMetadata

Saves image format and Metadata JSON format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SaveImageAndMetadata (
    string pathImage,
    string pathMetadata,
    Euresys.Open_eVision_2_10.EImageFileType type
)
```

Parameters

pathImage

The full path to the destination file.

pathMetadata

The full path to the destination file.

type

File format, as defined by [EImageFileType](#). If not specified, the file format is determined from the file extension.

EZMap.SaveMetadata

Saves Metadata in JSON format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SaveMetadata (
    string path
)
```

Parameters

path

The full path to the destination file.

EZMap.Serialize

Serializes the [EZMap](#) object with all its attributes.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Serialize (
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is read from or written to.

EZMap.SerializeImage

Serializes the image associated to [EZMap](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SerializeImage(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is read from or written to.

EZMap.SetBufferPtr

Sets the pointer to an externally allocated image buffer.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetBufferPtr(
    int width,
    int height,
    IntPtr imagePointer,
    int bitsPerRow
)
```

Parameters

width

The width of the supplied buffer, in pixels.

height

The height of the supplied buffer, in pixels.

imagePointer

The pointer (aligned on 4 bytes) to the buffer, which must be large enough to hold the data.

bitsPerRow

The total number of bits contained in a row, padding included.

Using the value **0** (default) means that this size is computed from the buffer width and the pixel size plus a padding with the smallest possible value that leads to a multiple of 4 bytes (32 bits), which is the minimum padding accepted by [EZMap::SetBufferPtr](#).

EZMap.SetResolution

Sets the resolution of the [EZMap](#) along the X, Y and Z axis.
On the Z axis, the resolution is the number of metric units per grey value.
On the X and Y axis, the resolution is the number of metric units per pixel.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetResolution(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint resolution
)
void SetResolution(
    float rx,
    float ry,
    float rz
)
```

Parameters

resolution

Contains the resolution along the X,Y and Z axis.

rx

Contains the resolution along the X axis.

ry

Contains the resolution along the Y axis.

rz

Contains the resolution along the Z axis.

EZMap.SetSize

Sets the width and height of the [EZMap](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetSize(
    int width,
    int height
)

void SetSize(
    Euresys.Open_eVision_2_10.Easy3D.EZMap other
)
```

Parameters

width

The new requested width.

height

The new requested height.

other

The other ZMap whose dimensions have to be used for the current object.

Remarks

Open eVision will allocate a new image buffer (deallocate the old image buffer) if the supplied width and height are different from the existing ones.

If an external buffer has been specified by means of **SetImagePtr**, it will be kept only if the size does not change.

Creating a new Open eVision image buffer and setting its size creates a 4-byte aligned buffer, by default. The *size of an ZMap* is specified as a number of columns (width) and rows (height).

The maximum image dimensions are 32767 by 32767. Furthermore, it must fit into the available memory, that depends upon the physical memory, the operating system and the memory already allocated by the process in other modules or libraries.

EZMap.Type

Pixel accessor type.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
abstract Euresys.Open_eVision_2_10.EImageType Type
{ get; }
```

EZMap.Width

Access ZMap Width.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
abstract int Width
{ get; set; }
```

EZMap.WorldShape

Returns the [EWorldShape](#) for conversion between 2D image and 2D world space coordinates. This EWorldShape can be used by EasyGauge to do measurements on a [EZMap](#) in real space coordinates (e.g mm).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
abstract Euresys.Open_eVision_2_10.EWorldShape WorldShape  
{ get; }
```

EZMap.WorldToImage

Transforms a 3D world position to a floating point (sub)pixel image position.

The image space origin is at the upper left corner of the image and the X/Y unit size is one pixel.

The Z value is given in grey scale value.

Returns TRUE if the pixel position is inside the image limits and the Z value is positive.

The parameter pixelPt is filled even if the point is outside the image.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
bool WorldToImage(  
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint worldPt,  
    ref Euresys.Open_eVision_2_10.Easy3D.E3DPoint pixelPt  
)
```

Parameters

worldPt

Position in the 3D world space.

pixelPt

Position in the image space.

EZMap.WorldToMapMatrix

Returns an [E3DTransformMatrix](#) that transforms positions from the world space to the [EZMap](#) space. This transformation is composed of rotation and translation only, so it is a rigid transformation and it preserves distances.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
abstract Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix WorldToMapMatrix
{ get; }
```

EZMap.WorldToZMap

Transforms a 3D world position to a 3D [EZMap](#) position.
The ZMap space origin is at the lower left corner of the image.
The scales of the world space and ZMap space are the same.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void WorldToZMap(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint worldPt,
    ref Euresys.Open_eVision_2_10.Easy3D.E3DPoint zmapPt
)
```

Parameters

worldPt

Position in the 3D world space.

zmapPt

Position in the ZMap space.

Remarks

Do not use this method with the same variable as input and output. It might lead to incorrect results.

EZMap.XResolution

Resolution of the [EZMap](#) along the X axis
The resolution is the number of metric units per pixel.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
abstract float XResolution  
  
    { get; set; }
```

EZMap.YResolution

Resolution of the [EZMap](#) along the Y axis
The resolution is the number of metric units per pixel.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
abstract float YResolution  
  
    { get; set; }
```

EZMap.ZMapToImage

Converts a 2D coordinate in the [EZMap](#) space to image (sub)pixel space.
(x,y) is the ZMap position (which has the same scale as the world space).
(u,v) is the corresponding pixel position (with its origin in the upper left corner of the image).
All values are expressed in floating point numbers.
Returns TRUE if the pixel position is inside the image limits.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool ZMapToImage (
    float x,
    float y,
    ref float u,
    ref float v
)
```

Parameters

- x*
Position along horizontal axis in the ZMap space.
- y*
Position along vertical axis in the ZMap space.
- u*
Column of the pixel as a floating point value.
- v*
Row of the pixel as a floating point value.

EZMap.ZMapToWorld

Transforms a 3D [EZMap](#) position to a 3D world space position.
The ZMap space origin is at the lower left corner of the image.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ZMapToWorld(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint zmapPt,
    ref Euresys.Open_eVision_2_10.Easy3D.E3DPoint worldPt
)
```

Parameters

zmapPt

Position in the ZMap space.

worldPt

Position in the 3D world space.

Remarks

Do not use this method with the same variable as input and output. It might lead to incorrect results.

EZMap.ZResolution

Resolution of the [EZMap](#) along the Z axis
The resolution is the number of grey values per pixel.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
abstract float ZResolution
{ get; set; }
```

4.195. EZMap16 Class

A ZMap16 is a 16bits corrected 2.5D image.

ZMap Pixel values (16 bits integers) represent distances from a 3D reference plane.

Distances are positive, during the ZMap generation all points below the reference plane are discarded.

The EZMap class also stores the transformation from the pixel coordinates to the real world coordinate system.

There could be undefined pixels in the ZMap.

Base Class: [EZMap](#)

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

Height

Access ZMap Height.

MapToWorldMatrix

Returns an [E3DTransformMatrix](#) that transforms positions from the [EZMap16](#) space to the world space. This transformation is composed of rotation and translation only, so it is a rigid transformation and preserves distances.

RowPitch

Returns the buffer row pitch.

Type

Pixel accessor type.

UndefinedValue

Returns the Undefined value. That value is used to set to mark pixels with no valid depth value.

Width

Access ZMap Width.

WorldShape

Returns the [EWorldShape](#) for conversion between 2D image and 2D world space coordinates. This [EWorldShape](#) can be used by [EasyGauge](#) to do measurements on a [EZMap16](#) in real space coordinates (e.g mm).

WorldToMapMatrix

Returns an [E3DTransformMatrix](#) that transforms positions from the world space to the [EZMap16](#) space. This transformation is composed of rotation and translation only, so it is a rigid transformation and it preserves distances.

XResolution

Resolution of the [EZMap16](#) along the X axis
The resolution is the number of metric units per pixel.

YResolution

Resolution of the [EZMap16](#) along the Y axis
The resolution is the number of metric units per pixel.

ZResolution

M Resolution of the [EZMap16](#) along the Z axis
The resolution is the number of grey values per pixel.

e

thods

AddMetadata

Adds a metadata key (name) and value.
If the metadata key already exists, its value will be overwritten.

AsEImage

Returns the [EZMap16](#) as an [EImageBW16](#) (16 bits gray scale) to use with existing [eVision](#) 2D tools.

Clear

Clears the ZMap: replaces all pixels with the undefined value.

ClearMetadata

Deletes all metadata.

ConvertCoordinatesMapToPixel

Converts 3D Map coordinates to Buffer coordinates

ConvertCoordinatesPixelToMap

Converts Buffer coordinates to 3D Map coordinates

CopyMetadataTo

Copies all metadata.

DeleteMetadata

Deletes value of this existing metadata key . Throws an exception if it does not exist.

Draw

Draws a [EZMap16](#) in a device context.

DrawImage

Displays the internal image buffer

EZMap16

Creates a 16 bits [EZMap](#).

FillUndefinedPixels

Fills undefined pixels, used to fill the "holes" in the ZMap.

GetBufferPtr

Clears the ZMap: replaces all pixels with the undefined value.

GetCheckedBufferPtr

Retrieves the pointer to the pixel buffer.

GetMetadata

Returns the string value of the given metadata.
Throws an exception if it does not exist.

GetPixel

Gets the value of a pixel .

GetPixelPositionFromWorldPosition

Returns in the u, v and value parameters the [EZMap16](#) values corresponding to a 3D world position.
The world position is projected on the ZMap reference plane to get a position in the ZMap.
If the projected position is outside the ZMap, the method returns FALSE.

GetResolution

Gets the resolution. For the X and Y axes, the resolution is expressed in metric units per pixel.
For the Z axis it is expressed in metric units per grey scale value.

GetSizeInWorld

Returns the dimensions of the [EZMap16](#) in real world space (e.g metric unit).

GetWorldPositionFromPixelPosition

Returns the 3D world position corresponding to a [EZMap16](#) pixel position. The world position is in the original point cloud space. The pixel space origin is at the upper left corner of the image and ranges to (width-1, height-1). The transformation to the world position is calculated using the center of the pixel. The pixel value at position (u,v) must not be undefined, otherwise an exception will be thrown.

GetZMapPositionFromPixelPosition

Returns the corresponding [EZMap16](#) 3D position of a ZMap pixel. The ZMap position is in the original point ZMap space. The pixel position origin is the upper left corner of the image and ranges to (width-1, height-1). The center of the pixel is used for the transformation to the ZMap position. Pixel at position (u,v) must be defined, otherwise an exception will be thrown.

GetZValue

Gets Z value (in metric coordinate) at pixel coordinates.

ImageToWorld

Transforms a floating point (sub)pixel image position to a 3D world position. The image space origin is at the upper left corner of the image and the X/Y unit size is one pixel. The pixel Z value is in grey scale values (its range depends on the ZMap type).

ImageToZMap

Converts a 2D image (sub)pixel coordinate to the [EZMap16](#) space. (u,v) is the pixel position (with its origin in the upper left corner of the image). (x,y) is the corresponding ZMap position (which has the same scale as the world space). All values are expressed in floating point numbers.

IsVoid

Tests if the [EZMap16](#) object size is zero.

Load

Restores the [EZMap16](#) stored in the given Open eVision file.

LoadImage

Restores the [EZMap16](#) image stored in the given image file.

LoadImageAndMetadata

Loads image format and Metadata in JSON format.

LoadMetadata

Loads Metadata in JSON format.

ModifyMetadata

Changes an existing metadata key and value. Throws an exception if it does not exist.

operator=

Assignment operator.

Save

Saves the [EZMap16](#) object to the given Open eVision file.

SaveImage

Saves the [EZMap16](#) image to the given image file.

SaveImageAndMetadata

Saves image format and Metadata JSON format.

SaveMetadata

Saves Metadata in JSON format.

Serialize

Serializes the [EZMap16](#) object with all its attributes.

SerializeImage

Serializes the image associated to [EZMap16](#).

SetBufferPtr

Sets the pointer to an externally allocated image buffer.

SetPixel

Sets the value of a pixel .

SetResolution

Sets the resolution. For the X and Y axes, the resolution is expressed in metric units per pixel.
For the Z axis it is expressed in metric units per grey scale value.

SetSize

Sets the width and height of the [EZMap16](#).

SetZValue

Sets Z value (in metric coordinate) at pixel coordinates.

WorldToImage

Transforms a 3D world position to a floating point (sub)pixel image position.
The image space origin is at the upper left corner of the image and the X/Y unit size is one pixel.
The Z value is given in grey scale value.
Returns TRUE if the pixel position is inside the image limits and the Z value is positive.
The parameter pixelPt is filled even if the point is outside the image.

WorldToZMap

Transforms a 3D world position to a 3D [EZMap16](#) position.
The ZMap space origin is at the lower left corner of the image.
The scales of the world space and ZMap space are the same.

ZMapToImage

Converts a 2D coordinate in the [EZMap16](#) space to image (sub-)pixel space.
(x,y) is the ZMap position (which has the same scale as the world space).
(u,v) is the corresponding pixel position (with its origin in the upper left corner of the image).
All values are expressed in floating point numbers.
Returns TRUE if the pixel position is inside the image limits.

ZMapToWorld

- Transforms a 3D [EZMap16](#) position to a 3D world space position.
- The ZMap space origin is at the lower left corner of the image.

Map16.AddMetadata

Adds a metadata key (name) and value.
If the metadata key already exists, its value will be overwritten.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void AddMetadata (
    string Key,
    string value
)
```

Parameters

Key

The name of the metadata. Names are unique.

value

The value for the given metadata.

EZMap16.AsEImage

Returns the [EZMap16](#) as an [EImageBW16](#) (16 bits gray scale) to use with existing eVision 2D tools.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.EImageBW16 AsEImage (
)
Euresys.Open_eVision_2_10.EImageBW16 AsEImage (
)
```

EZMap16.Clear

Clears the ZMap: replaces all pixels with the undefined value.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Clear (
)
```

EZMap16.ClearMetadata

Deletes all metadata.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ClearMetadata (
)
```

EZMap16.ConvertCoordinatesMapToPixel

Converts 3D Map coordinates to Buffer coordinates

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool ConvertCoordinatesMapToPixel (
    float x3D,
    float y3D,
    ref int xBuffer,
    ref int yBuffer
)
```

Parameters

x3D

The Map X coordinate.

y3D

The Map Y coordinate.

xBuffer

The returned Pixel X coordinate.

yBuffer

The returned Pixel Y coordinate.

EZMap16.ConvertCoordinatesPixelToMap

Converts Buffer coordinates to 3D Map coordinates

Namespace: Euresys.Open_eVision_2_10.Easy3D


```
[C#]
void ConvertCoordinatesPixelToMap(
    int xBuffer,
    int yBuffer,
    ref float x3D,
    ref float y3D
)
```

Parameters

xBuffer

The pixel X coordinate.

yBuffer

The pixel Y coordinate.

x3D

The returned Map X coordinate.

y3D

The returned Map Y coordinate.

EZMap16.CopyMetadataTo

Copies all metadata.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void CopyMetadataTo(
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 other
)
```

Parameters

other

An other [EZMap16](#).

EZMap16.DeleteMetadata

Deletes value of this existing metadata key . Throws an exception if it does not exist.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void DeleteMetadata (
    string Key
)
```

Parameters

Key

-

EZMap16.Draw

Draws a [EZMap16](#) in a device context.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Draw (
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

```

Parameters

graphicContext

Handle to the device context of the destination window.

zoomX

Magnification factor for zooming in or out in the horizontal direction. By default, the ZMap is displayed in 1:1 scale.

zoomY

Magnification factor for zooming in or out in the vertical direction. Setting a **0** value (which is the default) will result in isotropic scaling (i.e. equal horizontal and vertical factors).

panX

Pan offset (in pixels) in the horizontal direction. By default, no panning is applied.

panY

Pan offset (in pixels) in the vertical direction. By default, no panning is applied.

colorUndefinedPixel

An optional parameter to choose the drawing color of undefined pixels.

c24Vector

When supplied, this parameter allows using a LUT that maps from Depth to C24 when drawing (false colors).

bw8Vector

When supplied, this parameter allows using a LUT that maps from Depth to BW8 when drawing.

Remarks

A ZMap can be drawn (its pixels rendered) using a device context.

The horizontal and vertical zooming factors can be different but must be in the **1/16..16** range.

(MFC users can use the **CDC::GetSafeHdc()** method to obtain a suitable device context handle from a **CDC** instance.)

EZMap16.DrawImage

Displays the internal image buffer

Namespace: Euresys.Open_eVision_2_10.Easy3D

[C#]

```
void DrawImage(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void DrawImage(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void DrawImage(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```

void DrawImage(
    IntPtr graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

void DrawImage(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

void DrawImage(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

```

Parameters

graphicContext

Handle to the device context of the destination window.

zoomX

Magnification factor for zooming in or out in the horizontal direction. By default, the image is displayed in 1:1 scale.

zoomY

Magnification factor for zooming in or out in the vertical direction. Setting a **0** value (which is the default) will result in isotropic scaling (i.e. equal horizontal and vertical factors).

panX

Pan offset (in pixels) in the horizontal direction. By default, no panning is applied.

panY

Pan offset (in pixels) in the vertical direction. By default, no panning is applied.

colorUndefinedPixel

An optional parameter to choose the drawing color of undefined pixels.

c24Vector

When supplied, this parameter allows using a LUT that maps from Depth to C24 when drawing (false colors).

bw8Vector

When supplied, this parameter allows using a LUT that maps from Depth to BW8 when drawing.

Remarks

An image can be drawn (its pixels rendered) using a device context.

The horizontal and vertical zooming factors can be different but must be in the **1/16..16** range.

(MFC users can use the **CDC::GetSafeHdc()** method to obtain a suitable device context handle from a **CDC** instance.)

EZMap16.EZMap16

Creates a 16 bits [EZMap](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void EZMap16 (
)
void EZMap16 (
    int width,
    int height
)
void EZMap16 (
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 other
)
```

Parameters

width

The width of the new Zmap.

height

The height of the new Zmap.

other

Another Zmap.

EZMap16.FillUndefinedPixels

Fills undefined pixels, used to fill the "holes" in the ZMap.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void FillUndefinedPixels(
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 outMap,
    Euresys.Open_eVision_2_10.Easy3D.EFillUndefinedPixelsDirection direction,
    Euresys.Open_eVision_2_10.Easy3D.EFillUndefinedPixelsMethod method
)
```

Parameters

outMap

The destination ZMap.

direction

Direction in which the undefined pixels are filled in a ZMap from [EFillUndefinedPixelsDirection](#).

method

Which values used to fill the undefined pixels in a ZMap from [EFillUndefinedPixelsMethod](#).

EZMap16.GetBufferPtr

Clears the ZMap: replaces all pixels with the undefined value.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
IntPtr GetBufferPtr(
)
```

```
IntPtr GetBufferPtr(  
    int x,  
    int y  
)  
  
IntPtr GetBufferPtr(  
)  
  
IntPtr GetBufferPtr(  
    int x,  
    int y  
)
```

Parameters

x
-
y
-

EZMap16.GetCheckedBufferPtr

Retrieves the pointer to the pixel buffer.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
IntPtr GetCheckedBufferPtr(  
    int x,  
    int y  
)  
  
IntPtr GetCheckedBufferPtr(  
    int x,  
    int y  
)
```

Parameters

x

Column of the pixel of which we want the address.

y

Row of the pixel of which we want the address.

EZMap16.GetMetadata

Returns the string value of the given metadata.
Throws an exception if it does not exist.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
string GetMetadata(  
    string Key  
)
```

Parameters

Key

The name of an existing metadata.

EZMap16.GetPixel

Gets the value of a pixel .

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
Euresys.Open_eVision_2_10.EDepth16 GetPixel(  
    int x,  
    int y  
)
```

Parameters

- x*
Column of the pixel.
- y*
Row of the pixel.

EZMap16.GetPixelPositionFromWorldPosition

Returns in the *u*, *v* and *value* parameters the [EZMap16](#) values corresponding to a 3D world position. The world position is projected on the ZMap reference plane to get a position in the ZMap. If the projected position is outside the ZMap, the method returns FALSE.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
bool GetPixelPositionFromWorldPosition(  
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint world_position,  
    ref int u,  
    ref int v,  
    ref Euresys.Open_eVision_2_10.EDepth16 value  
)
```

Parameters

- world_position*
The 3D coordinates of a world position.
- u*
Column of the ZMap pixel in [0,width[.
- v*
Row of the ZMap pixel in [0,height[.

value

Value of the pixel.

EZMap16.GetResolution

Gets the resolution. For the X and Y axes, the resolution is expressed in metric units per pixel. For the Z axis it is expressed in metric units per grey scale value.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void GetResolution(
    ref float sx,
    ref float sy,
    ref float sz
)

Euresys.Open_eVision_2_10.Easy3D.E3DPoint GetResolution(
)
```

Parameters

sx

-

sy

-

sz

-

EZMap16.GetSizeInWorld

Returns the dimensions of the [EZMap16](#) in real world space (e.g metric unit).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void GetSizeInWorld(
    ref float worldWidth,
    ref float worldHeight
)
```

Parameters

worldWidth

Contains the size of the ZMap along the X axis (column).

worldHeight

Contains the size of the ZMap along the Y axis (row).

EZMap16.GetWorldPositionFromPixelPosition

Returns the 3D world position corresponding to a [EZMap16](#) pixel position. The world position is in the original point cloud space. The pixel space origin is at the upper left corner of the image and ranges to (width-1, height-1). The transformation to the world position is calculated using the center of the pixel. The pixel value at position (u,v) must not be undefined, otherwise an exception will be thrown.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DPoint GetWorldPositionFromPixelPosition(
    int u,
    int v
)
```

Parameters

u

Column of the pixel (bounds: [0,width]).

v

Row of the pixel (bounds: [0,height]).

EZMap16.GetZMapPositionFromPixelPosition

Returns the corresponding [EZMap16](#) 3D position of a ZMap pixel. The ZMap position is in the original point ZMap space. The pixel position origin is the upper left corner of the image and ranges to (width-1, height-1). The center of the pixel is used for the transformation to the ZMap position. Pixel at position (u,v) must be defined, otherwise an exception will be thrown.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DPoint GetZMapPositionFromPixelPosition(
    int u,
    int v
)
```

Parameters

u

Column of the pixel (bounds: [0,width]).

v

Row of the pixel (bounds: [0,height]).

EZMap16.GetZValue

Gets Z value (in metric coordinate) at pixel coordinates.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
float GetZValue(  
    int x,  
    int y  
)
```

Parameters

x
X Coordinate.

y
Y Coordinate.

EZMap16.Height

Access ZMap Height.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
override int Height  
  
    { get; set; }
```

EZMap16.ImageToWorld

Transforms a floating point (sub)pixel image position to a 3D world position. The image space origin is at the upper left corner of the image and the X/Y unit size is one pixel. The pixel Z value is in grey scale values (its range depends on the ZMap type).

Namespace: Euresys.Open_eVision_2_10.Easy3D


```
[C#]
void ImageToWorld(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint pixelPt,
    ref Euresys.Open_eVision_2_10.Easy3D.E3DPoint worldPt
)
```

Parameters

pixelPt

Position in the image space.

worldPt

Position in the 3D world space.

EZMap16.ImageToZMap

Converts a 2D image (sub)pixel coordinate to the [EZMap16](#) space.
(u,v) is the pixel position (with its origin in the upper left corner of the image).
(x,y) is the corresponding ZMap position (which has the same scale as the world space).
All values are expressed in floating point numbers.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ImageToZMap(
    float u,
    float v,
    ref float x,
    ref float y
)
```

Parameters

u

X Coordinate of the pixel as a floating point value.

v

Y Coordinate of the pixel as a floating point value.

x

Position along horizontal axis in the ZMap space.

y

Position along vertical axis in the ZMap space.

EZMap16.IsVoid

Tests if the [EZMap16](#) object size is zero.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
bool IsVoid(  
)
```

Remarks

Returns **TRUE** if the ZMap size is zero.

EZMap16.Load

Restores the [EZMap16](#) stored in the given Open eVision file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void Load(  
    string path  
)
```

Parameters

path

Full path to the file.

Remarks

When loading, the ZMap is resized if needed. This function restores all the ZMap attributes.

EZMap16.LoadImage

Restores the [EZMap16](#) image stored in the given image file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
void LoadImage(  
    string path  
)
```

Parameters

path

Full path to the file.

Remarks

When loading, the ZMap is resized if needed. This function does not restore the ZMap attributes, only the image associated with the [EZMap16](#) is updated.

EZMap16.LoadImageAndMetadata

Loads image format and Metadata in JSON format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
void LoadImageAndMetadata (  
    string path,  
    string pathMetadata  
)
```

Parameters

path

-

pathMetadata

Full path to the file.

EZMap16.LoadMetadata

Loads Metadata in JSON format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void LoadMetadata (  
    string path  
)
```

Parameters

path

Full path to the file.

EZMap16.MapToWorldMatrix

Returns an [EZ3DTransformMatrix](#) that transforms positions from the [EZMap16](#) space to the world space. This transformation is composed of rotation and translation only, so it is a rigid transformation and preserves distances.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix MapToWorldMatrix
    { get; }
```

EZMap16.ModifyMetadata

Changes an existing metadata key and value. Throws an exception if it does not exist.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ModifyMetadata(
    string Key,
    string value
)
```

Parameters

Key
-
value
-

EZMap16.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EZMap16 operator=(
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 other
)
```

Parameters

other

The source [EZMap16](#).

EZMap16.RowPitch

Returns the buffer row pitch.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override int RowPitch
{ get; }
```

EZMap16.Save

Saves the [EZMap16](#) object to the given Open eVision file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
void Save(  
    string path  
)
```

Parameters

path

The full path to the destination file.

Remarks

This format save the [EZMap16](#) in a Open eVision file. This function stores all the ZMap attributes.

EZMap16.SaveImage

Saves the [EZMap16](#) image to the given image file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
void SaveImage(  
    string path,  
    Euresys.Open_eVision_2_10.EImageFileType type  
)
```

Parameters

path

The full path to the destination file.

type

File format, as defined by [EImageFileType](#). If not specified, the file format is determined from the file extension.

Remarks

This format save the image associated to [EZMap16](#) in a standard image file and thus does not store ZMap attributes.

EZMap16.SaveImageAndMetadata

Saves image format and Metadata JSON format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SaveImageAndMetadata (
    string path,
    string pathMetadata,
    Euresys.Open_eVision_2_10.EImageFileType type
)
```

Parameters

- path*
-
- pathMetadata*
The full path to the destination file.
- type*
File format, as defined by [EImageFileType](#). If not specified, the file format is determined from the file extension.

EZMap16.SaveMetadata

Saves Metadata in JSON format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SaveMetadata (
    string path
)
```


Parameters

path

The full path to the destination file.

EZMap16.Serialize

Serializes the [EZMap16](#) object with all its attributes.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Serialize(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is read from or written to.

EZMap16.SerializeImage

Serializes the image associated to [EZMap16](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SerializeImage(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is read from or written to.

EZMap16.SetBufferPtr

Sets the pointer to an externally allocated image buffer.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetBufferPtr(
    int width,
    int height,
    IntPtr imagePointer,
    int bitsPerRow
)
```

Parameters

width

The width of the supplied buffer, in pixels.

height

The height of the supplied buffer, in pixels.

imagePointer

The pointer (aligned on 4 bytes) to the buffer, which must be large enough to hold the data.

bitsPerRow

The total number of bits contained in a row, padding included.

Using the value **0** (default) means that this size is computed from the buffer width and the pixel size plus a padding with the smallest possible value that leads to a multiple of 4 bytes (32 bits), which is the minimum padding accepted by [EZMap16::SetBufferPtr](#).

EZMap16.SetPixel

Sets the value of a pixel .

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetPixel(
    Euresys.Open_eVision_2_10.EDepth16 value,
    int x,
    int y
)
```

Parameters

- value*
Value of the pixel.
- x*
Column of the pixel.
- y*
Row of the pixel.

EZMap16.SetResolution

Sets the resolution. For the X and Y axes, the resolution is expressed in metric units per pixel. For the Z axis it is expressed in metric units per grey scale value.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetResolution(
    float rx,
    float ry,
    float rz
)

void SetResolution(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint resolution
)
```

Parameters

rx
-
ry
-
rz
-
resolution
-

EZMap16.SetSize

Sets the width and height of the [EZMap16](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetSize(
    int width,
    int height
)

void SetSize(
    Euresys.Open_eVision_2_10.Easy3D.EZMap other
)
```

Parameters

width
The new requested width.

height
The new requested height.

other
The other ZMap whose dimensions have to be used for the current object.

Remarks

Open eVision will allocate a new image buffer (deallocate the old image buffer) if the supplied width and height are different from the existing ones.

If an external buffer has been specified by means of **SetImagePtr**, it will be kept only if the size does not change. Creating a new Open eVision image buffer and setting its size creates a 4-byte aligned buffer, by default. The *size of an ZMap* is specified as a number of columns (width) and rows (height).

The maximum image dimensions are 32767 by 32767. Furthermore, it must fit into the available memory, that depends upon the physical memory, the operating system and the memory already allocated by the process in other modules or libraries.

EZMap16.SetZValue

Sets Z value (in metric coordinate) at pixel coordinates.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetZValue(
    float value,
    int x,
    int y
)
```

Parameters

value

Value of the pixel in metric space.

x

X Coordinate.

y

Y Coordinate.

EZMap16.Type

Pixel accessor type.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
override Euresys.Open_eVision_2_10.EImageType Type  
    { get; }
```

EZMap16.UndefinedValue

Returns the Undefined value. That value is used to set to mark pixels with no valid depth value.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.EDepth16 UndefinedValue  
    { get; }
```

EZMap16.Width

Access ZMap Width.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
override int Width  
    { get; set; }
```

EZMap16.WorldShape

Returns the [EWorldShape](#) for conversion between 2D image and 2D world space coordinates. This EWorldShape can be used by EasyGauge to do measurements on a [EZMap16](#) in real space coordinates (e.g mm).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override Euresys.Open_eVision_2_10.EWorldShape WorldShape
{ get; }
```

EZMap16.WorldToImage

Transforms a 3D world position to a floating point (sub)pixel image position.
The image space origin is at the upper left corner of the image and the X/Y unit size is one pixel.
The Z value is given in grey scale value.
Returns TRUE if the pixel position is inside the image limits and the Z value is positive.
The parameter pixelPt is filled even if the point is outside the image.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool WorldToImage (
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint worldPt,
    ref Euresys.Open_eVision_2_10.Easy3D.E3DPoint pixelPt
)
```

Parameters

worldPt

Position in the 3D world space.

pixelPt

Position in the image space.

EZMap16.WorldToMapMatrix

Returns an [E3DTransformMatrix](#) that transforms positions from the world space to the [EZMap16](#) space. This transformation is composed of rotation and translation only, so it is a rigid transformation and it preserves distances.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix WorldToMapMatrix
{ get; }
```

EZMap16.WorldToZMap

Transforms a 3D world position to a 3D [EZMap16](#) position.
The ZMap space origin is at the lower left corner of the image.
The scales of the world space and ZMap space are the same.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void WorldToZMap(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint worldPt,
    ref Euresys.Open_eVision_2_10.Easy3D.E3DPoint zmapPt
)
```

Parameters

worldPt

Position in the 3D world space.

zmapPt

Position in the ZMap space.

Remarks

Do not use this method with the same variable as input and output. It might lead to incorrect results.

EZMap16.XResolution

Resolution of the [EZMap16](#) along the X axis
The resolution is the number of metric units per pixel.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
override float XResolution  
  
    { get; set; }
```

EZMap16.YResolution

Resolution of the [EZMap16](#) along the Y axis
The resolution is the number of metric units per pixel.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
override float YResolution  
  
    { get; set; }
```

EZMap16.ZMapToImage

Converts a 2D coordinate in the [EZMap16](#) space to image (sub)pixel space.
(x,y) is the ZMap position (which has the same scale as the world space).
(u,v) is the corresponding pixel position (with its origin in the upper left corner of the image).
All values are expressed in floating point numbers.
Returns TRUE if the pixel position is inside the image limits.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool ZMapToImage (
    float x,
    float y,
    ref float u,
    ref float v
)
```

Parameters

- x*
Position along horizontal axis in the ZMap space.
- y*
Position along vertical axis in the ZMap space.
- u*
Column of the pixel as a floating point value.
- v*
Row of the pixel as a floating point value.

EZMap16.ZMapToWorld

Transforms a 3D [EZMap16](#) position to a 3D world space position.
The ZMap space origin is at the lower left corner of the image.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ZMapToWorld(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint zmapPt,
    ref Euresys.Open_eVision_2_10.Easy3D.E3DPoint worldPt
)
```

Parameters

zmapPt

Position in the ZMap space.

worldPt

Position in the 3D world space.

Remarks

Do not use this method with the same variable as input and output. It might lead to incorrect results.

EZMap16.ZResolution

Resolution of the [EZMap16](#) along the Z axis
The resolution is the number of grey values per pixel.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override float ZResolution
{ get; set; }
```

4.196. EZMap32f Class

A ZMap32f is a 32bits corrected 2.5D image.

ZMap pixel values (32 bits floating point numbers) represent distances from a 3D reference plane.

Distances are positive, during the ZMap generation all points below the reference plane are discarded.

The EZMap class also stores the transformation from the pixel coordinates to the real world coordinate system.

There could be undefined pixels in the ZMap.

Base Class: [EZMap](#)

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

[Height](#)

Access ZMap Height.

[MapToWorldMatrix](#)

Returns an [E3DTransformMatrix](#) that transforms positions from the [EZMap32f](#) space to the world space. This transformation is composed of rotation and translation only, so it is a rigid transformation and preserves distances.

[RowPitch](#)

Returns the buffer row pitch.

[Type](#)

Pixel accessor type.

[UndefinedValue](#)

Returns the Undefined value. That value is used to set to mark pixels with no valid depth value.

[Width](#)

Access ZMap Width.

WorldShape

Returns the [EWorldShape](#) for conversion between 2D image and 2D world space coordinates. This EWorldShape can be used by EasyGauge to do measurements on a [EZMap32f](#) in real space coordinates (e.g mm).

WorldToMapMatrix

Returns an [E3DTransformMatrix](#) that transforms positions from the world space to the [EZMap32f](#) space. This transformation is composed of rotation and translation only, so it is a rigid transformation and it preserves distances.

XResolution

Resolution of the [EZMap32f](#) along the X axis
The resolution is the number of metric units per pixel.

YResolution

Resolution of the [EZMap32f](#) along the Y axis
The resolution is the number of metric units per pixel.

ZResolution

M Resolution of the [EZMap32f](#) along the Z axis
The resolution is the number of grey values per pixel.

e

thods

AddMetadata

Adds a metadata key (name) and value.
If the metadata key already exists, its value will be overwritten.

AsEImage

Returns the [EZMap32f](#) as an [EImageBW32](#) (32 bits gray scale) to use with existing eVision 2D tools.

Clear

Clears the ZMap: replaces all pixels with the undefined value.

ClearMetadata

Deletes all metadata.

ConvertCoordinatesMapToPixel

Converts 3D Map coordinates to Buffer coordinates

ConvertCoordinatesPixelToMap

Converts Buffer coordinates to 3D Map coordinates

CopyMetadataTo

Copies all metadata.

DeleteMetadata

Deletes value of this existing metadata key . Throws an exception if it does not exist.

Draw

Draws a [EZMap32f](#) in a device context.

DrawImage

Displays the internal image buffer

EZMap32f

Creates a 32 bits [EZMap](#).

FillUndefinedPixels

Fills undefined pixels, used to fill the "holes" in the ZMap.

GetBufferPtr

Clears the ZMap: replaces all pixels with the undefined value.

GetCheckedBufferPtr

Retrieves the pointer to the pixel buffer.

GetMetadata

Returns the string value of the given metadata.
Throws an exception if it does not exist.

GetPixel

Gets the value of a pixel .

GetPixelPositionFromWorldPosition

Returns in the u, v and value parameters the [EZMap32f](#) values corresponding to a 3D world position.
The world position is projected on the ZMap reference plane to get a position in the ZMap.
If the projected position is outside the ZMap, the method returns FALSE.

GetResolution

Gets the resolution. For the X and Y axes, the resolution is expressed in metric units per pixel.
For the Z axis it is expressed in metric units per grey scale value.

GetSizeInWorld

Returns the dimensions of the [EZMap32f](#) in real world space (e.g metric unit).

GetWorldPositionFromPixelPosition

Returns the 3D world position corresponding to a [EZMap32f](#) pixel position. The world position is in the original point cloud space. The pixel space origin is at the upper left corner of the image and ranges to (width-1, height-1). The transformation to the world position is calculated using the center of the pixel. The pixel value at position (u,v) must not be undefined, otherwise an exception will be thrown.

GetZMapPositionFromPixelPosition

Returns the corresponding [EZMap32f](#) 3D position of a ZMap pixel. The ZMap position is in the original point ZMap space. The pixel position origin is the upper left corner of the image and ranges to (width-1, height-1). The center of the pixel is used for the transformation to the ZMap position. Pixel at position (u,v) must be defined, otherwise an exception will be thrown.

GetZValue

Gets Z value (in metric coordinate) at pixel coordinates.

ImageToWorld

Transforms a floating point (sub)pixel image position to a 3D world position. The image space origin is at the upper left corner of the image and the X/Y unit size is one pixel. The pixel Z value is in grey scale values (its range depends on the ZMap type).

ImageToZMap

Converts a 2D image (sub)pixel coordinate to the [EZMap32f](#) space. (u,v) is the pixel position (with its origin in the upper left corner of the image). (x,y) is the corresponding ZMap position (which has the same scale as the world space). All values are expressed in floating point numbers.

IsVoid

Tests if the [EZMap32f](#) object size is zero.

Load

Restores the [EZMap32f](#) stored in the given Open eVision file.

LoadImage

Restores the [EZMap32f](#) image stored in the given image file.

LoadImageAndMetadata

Loads image format and Metadata in JSON format.

LoadMetadata

Loads Metadata in JSON format.

ModifyMetadata

Changes an existing metadata key and value. Throws an exception if it does not exist.

operator=

Assignment operator.

Save

Saves the [EZMap32f](#) object to the given Open eVision file.

SaveImage

Saves the [EZMap32f](#) image to the given image file.

SaveImageAndMetadata

Saves image format and Metadata JSON format.

SaveMetadata

Saves Metadata in JSON format.

Serialize

Serializes the [EZMap32f](#) object with all its attributes.

SerializeImage

Serializes the image associated to [EZMap32f](#).

SetBufferPtr

Sets the pointer to an externally allocated image buffer.

SetPixel

Sets the value of a pixel .

SetResolution

Sets the resolution. For the X and Y axes, the resolution is expressed in metric units per pixel.
For the Z axis it is expressed in metric units per grey scale value.

SetSize

Sets the width and height of the [EZMap32f](#).

SetZValue

Sets Z value (in metric coordinate) at pixel coordinates.

WorldToImage

Transforms a 3D world position to a floating point (sub)pixel image position.
The image space origin is at the upper left corner of the image and the X/Y unit size is one pixel.
The Z value is given in grey scale value.
Returns TRUE if the pixel position is inside the image limits and the Z value is positive.
The parameter pixelPt is filled even if the point is outside the image.

WorldToZMap

Transforms a 3D world position to a 3D [EZMap32f](#) position.
The ZMap space origin is at the lower left corner of the image.
The scales of the world space and ZMap space are the same.

ZMapToImage

Converts a 2D coordinate in the [EZMap32f](#) space to image (sub-)pixel space.
(x,y) is the ZMap position (which has the same scale as the world space).
(u,v) is the corresponding pixel position (with its origin in the upper left corner of the image).
All values are expressed in floating point numbers.
Returns TRUE if the pixel position is inside the image limits.

ZMapToWorld

- └ Transforms a 3D [EZMap32f](#) position to a 3D world space position.
- └ The ZMap space origin is at the lower left corner of the image.

Map32f.AddMetadata

Adds a metadata key (name) and value.
If the metadata key already exists, its value will be overwritten.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void AddMetadata (
    string Key,
    string value
)
```

Parameters

Key

The name of the metadata. Names are unique.

value

The value for the given metadata.

EZMap32f.AsEImage

Returns the [EZMap32f](#) as an [EImageBW32](#) (32 bits gray scale) to use with existing eVision 2D tools.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.EImageBW32f AsEImage (
)
Euresys.Open_eVision_2_10.EImageBW32f AsEImage (
)
```

EZMap32f.Clear

Clears the ZMap: replaces all pixels with the undefined value.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Clear (
)
```

EZMap32f.ClearMetadata

Deletes all metadata.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ClearMetadata (
)
```

EZMap32f.ConvertCoordinatesMapToPixel

Converts 3D Map coordinates to Buffer coordinates

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool ConvertCoordinatesMapToPixel (
    float x3D,
    float y3D,
    ref int xBuffer,
    ref int yBuffer
)
```

Parameters

x3D

The Map X coordinate.

y3D

The Map Y coordinate.

xBuffer

The returned Pixel X coordinate.

yBuffer

The returned Pixel Y coordinate.

EZMap32f.ConvertCoordinatesPixelToMap

Converts Buffer coordinates to 3D Map coordinates

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ConvertCoordinatesPixelToMap(
    int xBuffer,
    int yBuffer,
    ref float x3D,
    ref float y3D
)
```

Parameters

xBuffer

The pixel X coordinate.

yBuffer

The pixel Y coordinate.

x3D

The returned Map X coordinate.

y3D

The returned Map Y coordinate.

EZMap32f.CopyMetadataTo

Copies all metadata.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void CopyMetadataTo(
    Euresys.Open_eVision_2_10.Easy3D.EZMap32f other
)
```

Parameters

other

An other [EZMap32f](#).

EZMap32f.DeleteMetadata

Deletes value of this existing metadata key . Throws an exception if it does not exist.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void DeleteMetadata(
    string Key
)
```

Parameters

Key

-

EZMap32f.Draw

Draws a [EZMap32f](#) in a device context.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```



```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

```

Parameters

graphicContext

Handle to the device context of the destination window.

zoomX

Magnification factor for zooming in or out in the horizontal direction. By default, the ZMap is displayed in 1:1 scale.

zoomY

Magnification factor for zooming in or out in the vertical direction. Setting a **0** value (which is the default) will result in isotropic scaling (i.e. equal horizontal and vertical factors).

panX

Pan offset (in pixels) in the horizontal direction. By default, no panning is applied.

panY

Pan offset (in pixels) in the vertical direction. By default, no panning is applied.

colorUndefinedPixel

An optional parameter to choose the drawing color of undefined pixels.

c24Vector

When supplied, this parameter allows using a LUT that maps from Depth to C24 when drawing (false colors).

bw8Vector

When supplied, this parameter allows using a LUT that maps from Depth to BW8 when drawing.

Remarks

A ZMap can be drawn (its pixels rendered) using a device context.

The horizontal and vertical zooming factors can be different but must be in the **1/16..16** range.

(MFC users can use the **CDC::GetSafeHdc()** method to obtain a suitable device context handle from a **CDC** instance.)

EZMap32f.DrawImage

Displays the internal image buffer

Namespace: Euresys.Open_eVision_2_10.Easy3D

[C#]

```
void DrawImage(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void DrawImage(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void DrawImage(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```

void DrawImage(
    IntPtr graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

void DrawImage(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

void DrawImage(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

```

Parameters

graphicContext

Handle to the device context of the destination window.

zoomX

Magnification factor for zooming in or out in the horizontal direction. By default, the image is displayed in 1:1 scale.

zoomY

Magnification factor for zooming in or out in the vertical direction. Setting a **0** value (which is the default) will result in isotropic scaling (i.e. equal horizontal and vertical factors).

panX

Pan offset (in pixels) in the horizontal direction. By default, no panning is applied.

panY

Pan offset (in pixels) in the vertical direction. By default, no panning is applied.

colorUndefinedPixel

An optional parameter to choose the drawing color of undefined pixels.

c24Vector

When supplied, this parameter allows using a LUT that maps from Depth to C24 when drawing (false colors).

bw8Vector

When supplied, this parameter allows using a LUT that maps from Depth to BW8 when drawing.

Remarks

An image can be drawn (its pixels rendered) using a device context.

The horizontal and vertical zooming factors can be different but must be in the **1/16..16** range.

(MFC users can use the **CDC::GetSafeHdc()** method to obtain a suitable device context handle from a **CDC** instance.)

EZMap32f.EZMap32f

Creates a 32 bits [EZMap](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void EZMap32f(
)
void EZMap32f(
    int width,
    int height
)
void EZMap32f(
    Euresys.Open_eVision_2_10.Easy3D.EZMap32f other
)
```

Parameters

width

The width of the new Zmap.

height

The height of the new Zmap.

other

Another Zmap.

EZMap32f.FillUndefinedPixels

Fills undefined pixels, used to fill the "holes" in the ZMap.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void FillUndefinedPixels (
    Euresys.Open_eVision_2_10.Easy3D.EZMap32f outMap,
    Euresys.Open_eVision_2_10.Easy3D.EFillUndefinedPixelsDirection direction,
    Euresys.Open_eVision_2_10.Easy3D.EFillUndefinedPixelsMethod method
)
```

Parameters

outMap

The destination ZMap.

direction

Direction in which the undefined pixels are filled in a ZMap from [EFillUndefinedPixelsDirection](#).

method

Which values used to fill the undefined pixels in a ZMap from [EFillUndefinedPixelsMethod](#).

EZMap32f.GetBufferPtr

Clears the ZMap: replaces all pixels with the undefined value.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
IntPtr GetBufferPtr (
)
```

```
IntPtr GetBufferPtr(  
    int x,  
    int y  
)  
  
IntPtr GetBufferPtr(  
)  
  
IntPtr GetBufferPtr(  
    int x,  
    int y  
)
```

Parameters

x
-
y
-

EZMap32f.GetCheckedBufferPtr

Retrieves the pointer to the pixel buffer.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
IntPtr GetCheckedBufferPtr(  
    int x,  
    int y  
)  
  
IntPtr GetCheckedBufferPtr(  
    int x,  
    int y  
)
```

Parameters

x

Column of the pixel of which we want the address.

y

Row of the pixel of which we want the address.

EZMap32f.GetMetadata

Returns the string value of the given metadata.
Throws an exception if it does not exist.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
string GetMetadata(  
    string Key  
)
```

Parameters

Key

The name of an existing metadata.

EZMap32f.GetPixel

Gets the value of a pixel .

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```



```
Euresys.Open_eVision_2_10.EDepth32f GetPixel(  
    int x,  
    int y  
)
```

Parameters

- x*
Column of the pixel.
- y*
Row of the pixel.

EZMap32f.GetPixelPositionFromWorldPosition

Returns in the *u*, *v* and *value* parameters the [EZMap32f](#) values corresponding to a 3D world position. The world position is projected on the ZMap reference plane to get a position in the ZMap. If the projected position is outside the ZMap, the method returns FALSE.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
bool GetPixelPositionFromWorldPosition(  
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint world_position,  
    ref int u,  
    ref int v,  
    ref Euresys.Open_eVision_2_10.EDepth32f value  
)
```

Parameters

- world_position*
The 3D coordinates of a world position.
- u*
Column of the ZMap pixel in [0,width[.
- v*
Row of the ZMap pixel in [0,height[.

value

Value of the pixel.

EZMap32f.GetResolution

Gets the resolution. For the X and Y axes, the resolution is expressed in metric units per pixel. For the Z axis it is expressed in metric units per grey scale value.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void GetResolution(
    ref float sx,
    ref float sy,
    ref float sz
)

Euresys.Open_eVision_2_10.Easy3D.E3DPoint GetResolution(
)
```

Parameters

sx

-

sy

-

sz

-

EZMap32f.GetSizeInWorld

Returns the dimensions of the [EZMap32f](#) in real world space (e.g metric unit).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void GetSizeInWorld(
    ref float worldWidth,
    ref float worldHeight
)
```

Parameters

worldWidth

Contains the size of the ZMap along the X axis (column).

worldHeight

Contains the size of the ZMap along the Y axis (row).

EZMap32f.GetWorldPositionFromPixelPosition

Returns the 3D world position corresponding to a [EZMap32f](#) pixel position. The world position is in the original point cloud space. The pixel space origin is at the upper left corner of the image and ranges to (width-1, height-1). The transformation to the world position is calculated using the center of the pixel. The pixel value at position (u,v) must not be undefined, otherwise an exception will be thrown.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DPoint GetWorldPositionFromPixelPosition(
    int u,
    int v
)
```

Parameters

u

Column of the pixel (bounds: [0,width]).

v

Row of the pixel (bounds: [0,height]).

EZMap32f.GetZMapPositionFromPixelPosition

Returns the corresponding [EZMap32f](#) 3D position of a ZMap pixel. The ZMap position is in the original point ZMap space. The pixel position origin is the upper left corner of the image and ranges to (width-1, height-1). The center of the pixel is used for the transformation to the ZMap position. Pixel at position (u,v) must be defined, otherwise an exception will be thrown.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DPoint GetZMapPositionFromPixelPosition(
    int u,
    int v
)
```

Parameters

u

Column of the pixel (bounds: [0,width]).

v

Row of the pixel (bounds: [0,height]).

EZMap32f.GetZValue

Gets Z value (in metric coordinate) at pixel coordinates.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
float GetZValue(  
    int x,  
    int y  
)
```

Parameters

x
X Coordinate.

y
Y Coordinate.

EZMap32f.Height

Access ZMap Height.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
override int Height  
  
    { get; set; }
```

EZMap32f.ImageToWorld

Transforms a floating point (sub)pixel image position to a 3D world position. The image space origin is at the upper left corner of the image and the X/Y unit size is one pixel. The pixel Z value is in grey scale values (its range depends on the ZMap type).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ImageToWorld(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint pixelPt,
    ref Euresys.Open_eVision_2_10.Easy3D.E3DPoint worldPt
)
```

Parameters

pixelPt

Position in the image space.

worldPt

Position in the 3D world space.

EZMap32f.ImageToZMap

Converts a 2D image (sub)pixel coordinate to the [EZMap32f](#) space.
(u,v) is the pixel position (with its origin in the upper left corner of the image).
(x,y) is the corresponding ZMap position (which has the same scale as the world space).
All values are expressed in floating point numbers.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ImageToZMap(
    float u,
    float v,
    ref float x,
    ref float y
)
```

Parameters

u

X Coordinate of the pixel as a floating point value.

v

Y Coordinate of the pixel as a floating point value.

x

Position along horizontal axis in the ZMap space.

y

Position along vertical axis in the ZMap space.

EZMap32f.IsVoid

Tests if the [EZMap32f](#) object size is zero.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool IsVoid(
)
```

Remarks

Returns **TRUE** if the ZMap size is zero.

EZMap32f.Load

Restores the [EZMap32f](#) stored in the given Open eVision file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Load(
    string path
)
```

Parameters

path

Full path to the file.

Remarks

When loading, the ZMap is resized if needed. This function restores all the ZMap attributes.

EZMap32f.LoadImage

Restores the [EZMap32f](#) image stored in the given image file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void LoadImage(
    string path
)
```

Parameters

path

Full path to the file.

Remarks

When loading, the ZMap is resized if needed. This function does not restore the ZMap attributes, only the image associated with the [EZMap32f](#) is updated.

EZMap32f.LoadImageAndMetadata

Loads image format and Metadata in JSON format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```



```
void LoadImageAndMetadata (  
    string path,  
    string pathMetadata  
)
```

Parameters

path

-

pathMetadata

Full path to the file.

EZMap32f.LoadMetadata

Loads Metadata in JSON format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void LoadMetadata (  
    string path  
)
```

Parameters

path

Full path to the file.

EZMap32f.MapToWorldMatrix

Returns an [EZ3DTransformMatrix](#) that transforms positions from the [EZMap32f](#) space to the world space. This transformation is composed of rotation and translation only, so it is a rigid transformation and preserves distances.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix MapToWorldMatrix
{ get; }
```

EZMap32f.ModifyMetadata

Changes an existing metadata key and value. Throws an exception if it does not exist.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ModifyMetadata(
    string Key,
    string value
)
```

Parameters

Key
-
value
-

EZMap32f.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EZMap32f operator=(
    Euresys.Open_eVision_2_10.Easy3D.EZMap32f other
)
```

Parameters

other

The source [EZMap32f](#).

EZMap32f.RowPitch

Returns the buffer row pitch.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override int RowPitch
{ get; }
```

EZMap32f.Save

Saves the [EZMap32f](#) object to the given Open eVision file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
void Save(  
    string path  
)
```

Parameters

path

The full path to the destination file.

Remarks

This format save the [EZMap32f](#) in a Open eVision file. This function stores all the ZMap attributes.

EZMap32f.SaveImage

Saves the [EZMap32f](#) image to the given image file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
void SaveImage(  
    string path,  
    Euresys.Open_eVision_2_10.EImageFileType type  
)
```

Parameters

path

The full path to the destination file.

type

File format, as defined by [EImageFileType](#). If not specified, the file format is determined from the file extension.

Remarks

This format save the image associated to [EZMap32f](#) in a standard image file and thus does not store ZMap attributes.

EZMap32f.SaveImageAndMetadata

Saves image format and Metadata JSON format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SaveImageAndMetadata (
    string path,
    string pathMetadata,
    Euresys.Open_eVision_2_10.EImageFileType type
)
```

Parameters

- path*
-
- pathMetadata*
- The full path to the destination file.
- type*
- File format, as defined by [EImageFileType](#). If not specified, the file format is determined from the file extension.

EZMap32f.SaveMetadata

Saves Metadata in JSON format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SaveMetadata (
    string path
)
```

Parameters

path

The full path to the destination file.

EZMap32f.Serialize

Serializes the [EZMap32f](#) object with all its attributes.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Serialize(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is read from or written to.

EZMap32f.SerializeImage

Serializes the image associated to [EZMap32f](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SerializeImage(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is read from or written to.

EZMap32f.SetBufferPtr

Sets the pointer to an externally allocated image buffer.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetBufferPtr(
    int width,
    int height,
    IntPtr imagePointer,
    int bitsPerRow
)
```

Parameters

width

The width of the supplied buffer, in pixels.

height

The height of the supplied buffer, in pixels.

imagePointer

The pointer (aligned on 4 bytes) to the buffer, which must be large enough to hold the data.

bitsPerRow

The total number of bits contained in a row, padding included.

Using the value **0** (default) means that this size is computed from the buffer width and the pixel size plus a padding with the smallest possible value that leads to a multiple of 4 bytes (32 bits), which is the minimum padding accepted by [EZMap32f::SetBufferPtr](#).

EZMap32f.SetPixel

Sets the value of a pixel .

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetPixel(
    Euresys.Open_eVision_2_10.EDepth32f value,
    int x,
    int y
)
```

Parameters

- value*
Value of the pixel.
- x*
Column of the pixel.
- y*
Row of the pixel.

EZMap32f.SetResolution

Sets the resolution. For the X and Y axes, the resolution is expressed in metric units per pixel. For the Z axis it is expressed in metric units per grey scale value.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetResolution(
    float rx,
    float ry,
    float rz
)

void SetResolution(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint resolution
)
```


Parameters

rx
-
ry
-
rz
-
resolution
-

EZMap32f.SetSize

Sets the width and height of the [EZMap32f](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetSize(
    int width,
    int height
)

void SetSize(
    Euresys.Open_eVision_2_10.Easy3D.EZMap other
)
```

Parameters

width
The new requested width.

height
The new requested height.

other
The other ZMap whose dimensions have to be used for the current object.

Remarks

Open eVision will allocate a new image buffer (deallocate the old image buffer) if the supplied width and height are different from the existing ones.

If an external buffer has been specified by means of **SetImagePtr**, it will be kept only if the size does not change. Creating a new Open eVision image buffer and setting its size creates a 4-byte aligned buffer, by default. The *size of an ZMap* is specified as a number of columns (width) and rows (height).

The maximum image dimensions are 32767 by 32767. Furthermore, it must fit into the available memory, that depends upon the physical memory, the operating system and the memory already allocated by the process in other modules or libraries.

EZMap32f.SetZValue

Sets Z value (in metric coordinate) at pixel coordinates.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetZValue (
    float value,
    int x,
    int y
)
```

Parameters

value

Value of the pixel in metric space.

x

X Coordinate.

y

Y Coordinate.

EZMap32f.Type

Pixel accessor type.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
override Euresys.Open_eVision_2_10.EImageType Type  
    { get; }
```

EZMap32f.UndefinedValue

Returns the Undefined value. That value is used to set to mark pixels with no valid depth value.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.EDepth32f UndefinedValue  
    { get; }
```

EZMap32f.Width

Access ZMap Width.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
override int Width  
    { get; set; }
```

EZMap32f.WorldShape

Returns the [EWorldShape](#) for conversion between 2D image and 2D world space coordinates. This EWorldShape can be used by EasyGauge to do measurements on a [EZMap32f](#) in real space coordinates (e.g mm).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override Euresys.Open_eVision_2_10.EWorldShape WorldShape
{ get; }
```

EZMap32f.WorldToImage

Transforms a 3D world position to a floating point (sub)pixel image position.
The image space origin is at the upper left corner of the image and the X/Y unit size is one pixel.
The Z value is given in grey scale value.
Returns TRUE if the pixel position is inside the image limits and the Z value is positive.
The parameter pixelPt is filled even if the point is outside the image.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool WorldToImage (
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint worldPt,
    ref Euresys.Open_eVision_2_10.Easy3D.E3DPoint pixelPt
)
```

Parameters

worldPt

Position in the 3D world space.

pixelPt

Position in the image space.

EZMap32f.WorldToMapMatrix

Returns an [E3DTransformMatrix](#) that transforms positions from the world space to the [EZMap32f](#) space. This transformation is composed of rotation and translation only, so it is a rigid transformation and it preserves distances.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix WorldToMapMatrix
{ get; }
```

EZMap32f.WorldToZMap

Transforms a 3D world position to a 3D [EZMap32f](#) position.
The ZMap space origin is at the lower left corner of the image.
The scales of the world space and ZMap space are the same.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void WorldToZMap(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint worldPt,
    ref Euresys.Open_eVision_2_10.Easy3D.E3DPoint zmapPt
)
```

Parameters

worldPt

Position in the 3D world space.

zmapPt

Position in the ZMap space.

Remarks

Do not use this method with the same variable as input and output. It might lead to incorrect results.

EZMap32f.XResolution

Resolution of the [EZMap32f](#) along the X axis
The resolution is the number of metric units per pixel.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
override float XResolution  
  
    { get; set; }
```

EZMap32f.YResolution

Resolution of the [EZMap32f](#) along the Y axis
The resolution is the number of metric units per pixel.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
override float YResolution  
  
    { get; set; }
```

EZMap32f.ZMapToImage

Converts a 2D coordinate in the [EZMap32f](#) space to image (sub)pixel space.
(x,y) is the ZMap position (which has the same scale as the world space).
(u,v) is the corresponding pixel position (with its origin in the upper left corner of the image).
All values are expressed in floating point numbers.
Returns TRUE if the pixel position is inside the image limits.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool ZMapToImage (
    float x,
    float y,
    ref float u,
    ref float v
)
```

Parameters

- x*
Position along horizontal axis in the ZMap space.
- y*
Position along vertical axis in the ZMap space.
- u*
Column of the pixel as a floating point value.
- v*
Row of the pixel as a floating point value.

EZMap32f.ZMapToWorld

Transforms a 3D [EZMap32f](#) position to a 3D world space position.
The ZMap space origin is at the lower left corner of the image.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ZMapToWorld(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint zmapPt,
    ref Euresys.Open_eVision_2_10.Easy3D.E3DPoint worldPt
)
```

Parameters

zmapPt

Position in the ZMap space.

worldPt

Position in the 3D world space.

Remarks

Do not use this method with the same variable as input and output. It might lead to incorrect results.

EZMap32f.ZResolution

Resolution of the [EZMap32f](#) along the Z axis
The resolution is the number of grey values per pixel.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override float ZResolution
{ get; set; }
```


4.197. EZMap8 Class

A ZMap8 is a 8bits corrected 2.5D image.

ZMap Pixel values (8 bits integers) represent distances from a 3D reference plane.

Distances are positive, during the ZMap generation all points below the reference plane are discarded.

The EZMap class also stores the transformation from the pixel coordinates to the real world coordinate system.

There could be undefined pixels in the ZMap.

Base Class: [EZMap](#)

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

[Height](#)

Access ZMap Height.

[MapToWorldMatrix](#)

Returns an [E3DTransformMatrix](#) that transforms positions from the [EZMap8](#) space to the world space. This transformation is composed of rotation and translation only, so it is a rigid transformation and preserves distances.

[RowPitch](#)

Returns the buffer row pitch.

[Type](#)

Pixel accessor type.

[UndefinedValue](#)

Returns the Undefined value. That value is used to set to mark pixels with no valid depth value.

[Width](#)

Access ZMap Width.

WorldShape

Returns the [EWorldShape](#) for conversion between 2D image and 2D world space coordinates. This [EWorldShape](#) can be used by [EasyGauge](#) to do measurements on a [EZMap8](#) in real space coordinates (e.g mm).

WorldToMapMatrix

Returns an [E3DTransformMatrix](#) that transforms positions from the world space to the [EZMap8](#) space. This transformation is composed of rotation and translation only, so it is a rigid transformation and it preserves distances.

XResolution

Resolution of the [EZMap8](#) along the X axis
The resolution is the number of metric units per pixel.

YResolution

Resolution of the [EZMap8](#) along the Y axis
The resolution is the number of metric units per pixel.

ZResolution

M Resolution of the [EZMap8](#) along the Z axis
The resolution is the number of grey values per pixel.

e

thods

AddMetadata

Adds a metadata key (name) and value.
If the metadata key already exists, its value will be overwritten.

AsEImage

Returns the [EZMap8](#) as an [EImageBW8](#) (8 bits gray scale) to use with existing [eVision](#) 2D tools.

Clear

Clears the ZMap: replaces all pixels with the undefined value.

ClearMetadata

Deletes all metadata.

ConvertCoordinatesMapToPixel

Converts 3D Map coordinates to Buffer coordinates

ConvertCoordinatesPixelToMap

Converts Buffer coordinates to 3D Map coordinates

CopyMetadataTo

Copies all metadata.

DeleteMetadata

Deletes value of this existing metadata key . Throws an exception if it does not exist.

Draw

Draws a [EZMap8](#) in a device context.

DrawImage

Displays the internal image buffer

EZMap8

Creates a 8 bits [EZMap](#).

FillUndefinedPixels

Fills undefined pixels, used to fill the "holes" in the ZMap.

GetBufferPtr

Retrieves the pointer to the internal pixel buffer.

GetCheckedBufferPtr

Retrieves the pointer to the pixel buffer.

GetMetadata

Returns the string value of the given metadata.
Throws an exception if it does not exist.

GetPixel

Gets the value of a pixel .

GetPixelPositionFromWorldPosition

Returns in the u, v and value parameters the [EZMap8](#) values corresponding to a 3D world position.
The world position is projected on the ZMap reference plane to get a position in the ZMap.
If the projected position is outside the ZMap, the method returns FALSE.

GetResolution

Gets the resolution. For the X and Y axes, the resolution is expressed in metric units per pixel.
For the Z axis it is expressed in metric units per grey scale value.

GetSizeInWorld

Returns the dimensions of the [EZMap8](#) in real world space (e.g metric unit).

GetWorldPositionFromPixelPosition

Returns the 3D world position corresponding to a [EZMap8](#) pixel position. The world position is in the original point cloud space. The pixel space origin is at the upper left corner of the image and ranges to (width-1, height-1). The transformation to the world position is calculated using the center of the pixel. The pixel value at position (u,v) must not be undefined, otherwise an exception will be thrown.

GetZMapPositionFromPixelPosition

Returns the corresponding [EZMap8](#) 3D position of a ZMap pixel. The ZMap position is in the original point ZMap space. The pixel position origin is the upper left corner of the image and ranges to (width-1, height-1). The center of the pixel is used for the transformation to the ZMap position. Pixel at position (u,v) must be defined, otherwise an exception will be thrown.

GetZValue

Gets Z value (in metric coordinate) at pixel coordinates.

ImageToWorld

Transforms a floating point (sub)pixel image position to a 3D world position. The image space origin is at the upper left corner of the image and the X/Y unit size is one pixel. The pixel Z value is in grey scale values (its range depends on the ZMap type).

ImageToZMap

Converts a 2D image (sub)pixel coordinate to the [EZMap8](#) space. (u,v) is the pixel position (with its origin in the upper left corner of the image). (x,y) is the corresponding ZMap position (which has the same scale as the world space). All values are expressed in floating point numbers.

IsVoid

Tests if the [EZMap8](#) object size is zero.

Load

Restores the [EZMap8](#) stored in the given Open eVision file.

LoadImage

Restores the [EZMap8](#) image stored in the given image file.

LoadImageAndMetadata

Loads image format and Metadata in JSON format.

LoadMetadata

Loads Metadata in JSON format.

ModifyMetadata

Changes an existing metadata key and value. Throws an exception if it does not exist.

operator=

Assignment operator.

Save

Saves the [EZMap8](#) object to the given Open eVision file.

SaveImage

Saves the [EZMap8](#) image to the given image file.

SaveImageAndMetadata

Saves image format and Metadata JSON format.

SaveMetadata

Saves Metadata in JSON format.

Serialize

Serializes the [EZMap8](#) object with all its attributes.

SerializeImage

Serializes the image associated to [EZMap8](#).

SetBufferPtr

Sets the pointer to an externally allocated image buffer.

SetPixel

Sets the value of a pixel .

SetResolution

Sets the resolution. For the X and Y axes, the resolution is expressed in metric units per pixel.
For the Z axis it is expressed in metric units per grey scale value.

SetSize

Sets the width and height of the [EZMap8](#).

SetZValue

Sets Z value (in metric coordinate) at pixel coordinates.

WorldToImage

Transforms a 3D world position to a floating point (sub)pixel image position.
The image space origin is at the upper left corner of the image and the X/Y unit size is one pixel.
The Z value is given in grey scale value.
Returns TRUE if the pixel position is inside the image limits and the Z value is positive.
The parameter pixelPt is filled even if the point is outside the image.

WorldToZMap

Transforms a 3D world position to a 3D [EZMap8](#) position.
The ZMap space origin is at the lower left corner of the image.
The scales of the world space and ZMap space are the same.

ZMapToImage

Converts a 2D coordinate in the [EZMap8](#) space to image (sub)pixel space.
(x,y) is the ZMap position (which has the same scale as the world space).
(u,v) is the corresponding pixel position (with its origin in the upper left corner of the image).
All values are expressed in floating point numbers.
Returns TRUE if the pixel position is inside the image limits.

ZMapToWorld

- Transforms a 3D EZMap8 position to a 3D world space position.
- The ZMap space origin is at the lower left corner of the image.

Map8.AddMetadata

Adds a metadata key (name) and value.
If the metadata key already exists, its value will be overwritten.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
void AddMetadata (  
    string Key,  
    string value  
)
```

Parameters

Key

The name of the metadata. Names are unique.

value

The value for the given metadata.

EZMap8.AsEImage

Returns the EZMap8 as an EImageBW8 (8 bits gray scale) to use with existing eVision 2D tools.

Namespace: Euresys.Open_eVision_2_10.Easy3D


```
[C#]
Euresys.Open_eVision_2_10.EImageBW8 AsEImage (
)
Euresys.Open_eVision_2_10.EImageBW8 AsEImage (
)
```

EZMap8.Clear

Clears the ZMap: replaces all pixels with the undefined value.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Clear (
)
```

EZMap8.ClearMetadata

Deletes all metadata.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ClearMetadata (
)
```

EZMap8.ConvertCoordinatesMapToPixel

Converts 3D Map coordinates to Buffer coordinates

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool ConvertCoordinatesMapToPixel (
    float x3D,
    float y3D,
    ref int xBuffer,
    ref int yBuffer
)
```

Parameters

x3D

The Map X coordinate.

y3D

The Map Y coordinate.

xBuffer

The returned Pixel X coordinate.

yBuffer

The returned Pixel Y coordinate.

EZMap8.ConvertCoordinatesPixelToMap

Converts Buffer coordinates to 3D Map coordinates

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ConvertCoordinatesPixelToMap(
    int xBuffer,
    int yBuffer,
    ref float x3D,
    ref float y3D
)
```

Parameters

xBuffer

The pixel X coordinate.

yBuffer

The pixel Y coordinate.

x3D

The returned Map X coordinate.

y3D

The returned Map Y coordinate.

EZMap8.CopyMetadataTo

Copies all metadata.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void CopyMetadataTo(
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 other
)
```

Parameters

other

An other [EZMap8](#).

EZMap8.DeleteMetadata

Deletes value of this existing metadata key . Throws an exception if it does not exist.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void DeleteMetadata(
    string Key
)
```

Parameters

Key

-

EZMap8.Draw

Draws a [EZMap8](#) in a device context.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Draw(
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY  
)
```

```
void Draw(  
    IntPtr graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY
)

void Draw(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

```

Parameters

graphicContext

Handle to the device context of the destination window.

zoomX

Magnification factor for zooming in or out in the horizontal direction. By default, the ZMap is displayed in 1:1 scale.

zoomY

Magnification factor for zooming in or out in the vertical direction. Setting a **0** value (which is the default) will result in isotropic scaling (i.e. equal horizontal and vertical factors).

panX

Pan offset (in pixels) in the horizontal direction. By default, no panning is applied.

panY

Pan offset (in pixels) in the vertical direction. By default, no panning is applied.

colorUndefinedPixel

An optional parameter to choose the drawing color of undefined pixels.

c24Vector

When supplied, this parameter allows using a LUT that maps from Depth to C24 when drawing (false colors).

bw8Vector

When supplied, this parameter allows using a LUT that maps from Depth to BW8 when drawing.

Remarks

A ZMap can be drawn (its pixels rendered) using a device context.

The horizontal and vertical zooming factors can be different but must be in the **1/16..16** range.

(MFC users can use the **CDC::GetSafeHdc()** method to obtain a suitable device context handle from a **CDC** instance.)

EZMap8.DrawImage

Displays the internal image buffer

Namespace: Euresys.Open_eVision_2_10.Easy3D

[C#]

```
void DrawImage(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void DrawImage(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```

```
void DrawImage(  
    Euresys.Open_eVision_2_10.EDrawAdapter graphicContext,  
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,  
    float zoomX,  
    float zoomY,  
    float panX,  
    float panY,  
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel  
)
```



```

void DrawImage(
    IntPtr graphicContext,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

void DrawImage(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EC24Vector c24Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

void DrawImage(
    IntPtr graphicContext,
    Euresys.Open_eVision_2_10.EBW8Vector bw8Vector,
    float zoomX,
    float zoomY,
    float panX,
    float panY,
    Euresys.Open_eVision_2_10.EC24 colorUndefinedPixel
)

```

Parameters

graphicContext

Handle to the device context of the destination window.

zoomX

Magnification factor for zooming in or out in the horizontal direction. By default, the image is displayed in 1:1 scale.

zoomY

Magnification factor for zooming in or out in the vertical direction. Setting a **0** value (which is the default) will result in isotropic scaling (i.e. equal horizontal and vertical factors).

panX

Pan offset (in pixels) in the horizontal direction. By default, no panning is applied.

panY

Pan offset (in pixels) in the vertical direction. By default, no panning is applied.

colorUndefinedPixel

An optional parameter to choose the drawing color of undefined pixels.

c24Vector

When supplied, this parameter allows using a LUT that maps from Depth to C24 when drawing (false colors).

bw8Vector

When supplied, this parameter allows using a LUT that maps from Depth to BW8 when drawing.

Remarks

An image can be drawn (its pixels rendered) using a device context.

The horizontal and vertical zooming factors can be different but must be in the **1/16..16** range.

(MFC users can use the **CDC::GetSafeHdc()** method to obtain a suitable device context handle from a **CDC** instance.)

EZMap8.EZMap8

Creates a 8 bits [EZMap](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void EZMap8 (
)
void EZMap8 (
    int width,
    int height
)
void EZMap8 (
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 other
)
```

Parameters

width

The width of the new Zmap.

height

The height of the new Zmap.

other

Another Zmap.

EZMap8.FillUndefinedPixels

Fills undefined pixels, used to fill the "holes" in the ZMap.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void FillUndefinedPixels(
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 outMap,
    Euresys.Open_eVision_2_10.Easy3D.EFillUndefinedPixelsDirection direction,
    Euresys.Open_eVision_2_10.Easy3D.EFillUndefinedPixelsMethod method
)
```

Parameters

outMap

The destination ZMap.

direction

Direction in which the undefined pixels are filled in a ZMap from [EFillUndefinedPixelsDirection](#).

method

Which values used to fill the undefined pixels in a ZMap from [EFillUndefinedPixelsMethod](#).

EZMap8.GetBufferPtr

Retrieves the pointer to the internal pixel buffer.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
IntPtr GetBufferPtr(
)
```

```
IntPtr GetBufferPtr(  
    int x,  
    int y  
)  
  
IntPtr GetBufferPtr(  
)  
  
IntPtr GetBufferPtr(  
    int x,  
    int y  
)
```

Parameters

x

Column of the pixel which we want the address.

y

Row of the pixel which we want the address.

Remarks

This function does not check the value of the parameters.
Use carefully.

EZMap8.GetCheckedBufferPtr

Retrieves the pointer to the pixel buffer.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
IntPtr GetCheckedBufferPtr(  
    int x,  
    int y  
)
```

```
IntPtr GetCheckedBufferPtr (
    int x,
    int y
)
```

Parameters

- x*
Column of the pixel of which we want the address.
- y*
Row of the pixel of which we want the address.

EZMap8.GetMetadata

Returns the string value of the given metadata.
Throws an exception if it does not exist.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
string GetMetadata (
    string Key
)
```

Parameters

- Key*
The name of an existing metadata.

EZMap8.GetPixel

Gets the value of a pixel .

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.EDepth8 GetPixel(
    int x,
    int y
)
```

Parameters

- x*
Column of the pixel.
- y*
Row of the pixel.

EZMap8.GetPixelPositionFromWorldPosition

Returns in the *u*, *v* and *value* parameters the [EZMap8](#) values corresponding to a 3D world position. The world position is projected on the ZMap reference plane to get a position in the ZMap. If the projected position is outside the ZMap, the method returns FALSE.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool GetPixelPositionFromWorldPosition(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint world_position,
    ref int u,
    ref int v,
    ref Euresys.Open_eVision_2_10.EDepth8 value
)
```

Parameters

- world_position*
The 3D coordinates of a world position.
- u*
Column of the ZMap pixel in [0,width[.
- v*

Row of the ZMap pixel in [0,height[.

value

Value of the pixel.

EZMap8.GetResolution

Gets the resolution. For the X and Y axes, the resolution is expressed in metric units per pixel. For the Z axis it is expressed in metric units per grey scale value.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void GetResolution(
    ref float sx,
    ref float sy,
    ref float sz
)
Euresys.Open_eVision_2_10.Easy3D.E3DPoint GetResolution(
)
```

Parameters

sx

-

sy

-

sz

-

EZMap8.GetSizeInWorld

Returns the dimensions of the [EZMap8](#) in real world space (e.g metric unit).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void GetSizeInWorld(
    ref float worldWidth,
    ref float worldHeight
)
```

Parameters

worldWidth

Contains the size of the ZMap along the X axis (column).

worldHeight

Contains the size of the ZMap along the Y axis (row).

EZMap8.GetWorldPositionFromPixelPosition

Returns the 3D world position corresponding to a [EZMap8](#) pixel position. The world position is in the original point cloud space. The pixel space origin is at the upper left corner of the image and ranges to (width-1, height-1). The transformation to the world position is calculated using the center of the pixel. The pixel value at position (u,v) must not be undefined, otherwise an exception will be thrown.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DPoint GetWorldPositionFromPixelPosition(
    int u,
    int v
)
```

Parameters

u

Column of the pixel (bounds: [0,width]).

v

Row of the pixel (bounds: [0,height]).

EZMap8.GetZMapPositionFromPixelPosition

Returns the corresponding [EZMap8](#) 3D position of a ZMap pixel. The ZMap position is in the original point ZMap space. The pixel position origin is the upper left corner of the image and ranges to (width-1, height-1). The center of the pixel is used for the transformation to the ZMap position. Pixel at position (u,v) must be defined, otherwise an exception will be thrown.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.E3DPoint GetZMapPositionFromPixelPosition(
    int u,
    int v
)
```

Parameters

u

Column of the pixel (bounds: [0,width]).

v

Row of the pixel (bounds: [0,height]).

EZMap8.GetZValue

Gets Z value (in metric coordinate) at pixel coordinates.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
float GetZValue(  
    int x,  
    int y  
)
```

Parameters

x
X Coordinate.

y
Y Coordinate.

EZMap8.Height

Access ZMap Height.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
override int Height  
  
    { get; set; }
```

EZMap8.ImageToWorld

Transforms a floating point (sub)pixel image position to a 3D world position. The image space origin is at the upper left corner of the image and the X/Y unit size is one pixel. The pixel Z value is in grey scale values (its range depends on the ZMap type).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ImageToWorld(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint pixelPt,
    ref Euresys.Open_eVision_2_10.Easy3D.E3DPoint worldPt
)
```

Parameters

pixelPt

Position in the image space.

worldPt

Position in the 3D world space.

EZMap8.ImageToZMap

Converts a 2D image (sub)pixel coordinate to the [EZMap8](#) space.

(u,v) is the pixel position (with its origin in the upper left corner of the image).

(x,y) is the corresponding ZMap position (which has the same scale as the world space).

All values are expressed in floating point numbers.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ImageToZMap(
    float u,
    float v,
    ref float x,
    ref float y
)
```

Parameters

u

X Coordinate of the pixel as a floating point value.

v

Y Coordinate of the pixel as a floating point value.

x

Position along horizontal axis in the ZMap space.

y

Position along vertical axis in the ZMap space.

EZMap8.IsVoid

Tests if the [EZMap8](#) object size is zero.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool IsVoid(
)
```

Remarks

Returns **TRUE** if the ZMap size is zero.

EZMap8.Load

Restores the [EZMap8](#) stored in the given Open eVision file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Load(
    string path
)
```

Parameters

path

Full path to the file.

Remarks

When loading, the ZMap is resized if needed. This function restores all the ZMap attributes.

EZMap8.LoadImage

Restores the [EZMap8](#) image stored in the given image file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void LoadImage(
    string path
)
```

Parameters

path

Full path to the file.

Remarks

When loading, the ZMap is resized if needed. This function does not restore the ZMap attributes, only the image associated with the [EZMap8](#) is updated.

EZMap8.LoadImageAndMetadata

Loads image format and Metadata in JSON format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
void LoadImageAndMetadata (  
    string pathImage,  
    string pathMetadata  
)
```

Parameters

pathImage

Full path to the file.

pathMetadata

Full path to the file.

EZMap8.LoadMetadata

Loads Metadata in JSON format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void LoadMetadata (  
    string path  
)
```

Parameters

path

Full path to the file.

EZMap8.MapToWorldMatrix

Returns an [EZ3DTransformMatrix](#) that transforms positions from the [EZMap8](#) space to the world space. This transformation is composed of rotation and translation only, so it is a rigid transformation and preserves distances.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix MapToWorldMatrix
{ get; }
```

EZMap8.ModifyMetadata

Changes an existing metadata key and value. Throws an exception if it does not exist.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ModifyMetadata (
    string Key,
    string value
)
```

Parameters

Key
-
value
-

EZMap8.operator=

Assignment operator.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
Euresys.Open_eVision_2_10.Easy3D.EZMap8 operator=(
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 other
)
```

Parameters

other

The source [EZMap8](#).

EZMap8.RowPitch

Returns the buffer row pitch.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override int RowPitch
{ get; }
```

EZMap8.Save

Saves the [EZMap8](#) object to the given Open eVision file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```



```
void Save(  
    string path  
)
```

Parameters

path

The full path to the destination file.

Remarks

This format save the [EZMap8](#) in a Open eVision file. This function stores all the ZMap attributes.

EZMap8.SaveImage

Saves the [EZMap8](#) image to the given image file.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
  
void SaveImage(  
    string path,  
    Euresys.Open_eVision_2_10.EImageFileType type  
)
```

Parameters

path

The full path to the destination file.

type

File format, as defined by [EImageFileType](#). If not specified, the file format is determined from the file extension.

Remarks

This format save the image associated to [EZMap8](#) in a standard image file and thus does not store ZMap attributes.

EZMap8.SaveImageAndMetadata

Saves image format and Metadata JSON format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SaveImageAndMetadata (
    string pathImage,
    string pathMetadata,
    Euresys.Open_eVision_2_10.EImageFileType type
)
```

Parameters

pathImage

The full path to the destination file.

pathMetadata

The full path to the destination file.

type

File format, as defined by [EImageFileType](#). If not specified, the file format is determined from the file extension.

EZMap8.SaveMetadata

Saves Metadata in JSON format.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SaveMetadata (
    string path
)
```

Parameters

path

The full path to the destination file.

EZMap8.Serialize

Serializes the [EZMap8](#) object with all its attributes.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Serialize(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is read from or written to.

EZMap8.SerializeImage

Serializes the image associated to [EZMap8](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SerializeImage(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is read from or written to.

EZMap8.SetBufferPtr

Sets the pointer to an externally allocated image buffer.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetBufferPtr(
    int width,
    int height,
    IntPtr imagePointer,
    int bitsPerRow
)
```

Parameters

width

The width of the supplied buffer, in pixels.

height

The height of the supplied buffer, in pixels.

imagePointer

The pointer (aligned on 4 bytes) to the buffer, which must be large enough to hold the data.

bitsPerRow

The total number of bits contained in a row, padding included.

Using the value **0** (default) means that this size is computed from the buffer width and the pixel size plus a padding with the smallest possible value that leads to a multiple of 4 bytes (32 bits), which is the minimum padding accepted by [EZMap8::SetBufferPtr](#).

EZMap8.SetPixel

Sets the value of a pixel .

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetPixel(
    Euresys.Open_eVision_2_10.EDepth8 value,
    int x,
    int y
)
```

Parameters

- value*
Value of the pixel.
- x*
Column of the pixel.
- y*
Row of the pixel.

EZMap8.SetResolution

Sets the resolution. For the X and Y axes, the resolution is expressed in metric units per pixel. For the Z axis it is expressed in metric units per grey scale value.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetResolution(
    float rx,
    float ry,
    float rz
)

void SetResolution(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint resolution
)
```

Parameters

rx
-
ry
-
rz
-
resolution
-

EZMap8.SetSize

Sets the width and height of the [EZMap8](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetSize(
    int width,
    int height
)

void SetSize(
    Euresys.Open_eVision_2_10.Easy3D.EZMap other
)
```

Parameters

width
The new requested width.

height
The new requested height.

other
The other ZMap whose dimensions have to be used for the current object.

Remarks

Open eVision will allocate a new image buffer (deallocate the old image buffer) if the supplied width and height are different from the existing ones.

If an external buffer has been specified by means of **SetImagePtr**, it will be kept only if the size does not change. Creating a new Open eVision image buffer and setting its size creates a 4-byte aligned buffer, by default. The *size of an ZMap* is specified as a number of columns (width) and rows (height).

The maximum image dimensions are 32767 by 32767. Furthermore, it must fit into the available memory, that depends upon the physical memory, the operating system and the memory already allocated by the process in other modules or libraries.

EZMap8.SetZValue

Sets Z value (in metric coordinate) at pixel coordinates.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void SetZValue(
    float value,
    int x,
    int y
)
```

Parameters

value

Value of the pixel in metric space.

x

X Coordinate.

y

Y Coordinate.

EZMap8.Type

Pixel accessor type.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
override Euresys.Open_eVision_2_10.EImageType Type  
    { get; }
```

EZMap8.UndefinedValue

Returns the Undefined value. That value is used to set to mark pixels with no valid depth value.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.EDepth8 UndefinedValue  
    { get; }
```

EZMap8.Width

Access ZMap Width.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
override int Width  
    { get; set; }
```


EZMap8.WorldShape

Returns the [EWorldShape](#) for conversion between 2D image and 2D world space coordinates. This EWorldShape can be used by EasyGauge to do measurements on a [EZMap8](#) in real space coordinates (e.g mm).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override Euresys.Open_eVision_2_10.EWorldShape WorldShape
{ get; }
```

EZMap8.WorldToImage

Transforms a 3D world position to a floating point (sub)pixel image position.
The image space origin is at the upper left corner of the image and the X/Y unit size is one pixel.
The Z value is given in grey scale value.
Returns TRUE if the pixel position is inside the image limits and the Z value is positive.
The parameter pixelPt is filled even if the point is outside the image.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool WorldToImage (
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint worldPt,
    ref Euresys.Open_eVision_2_10.Easy3D.E3DPoint pixelPt
)
```

Parameters

worldPt

Position in the 3D world space.

pixelPt

Position in the image space.

EZMap8.WorldToMapMatrix

Returns an [E3DTransformMatrix](#) that transforms positions from the world space to the [EZMap8](#) space. This transformation is composed of rotation and translation only, so it is a rigid transformation and it preserves distances.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override Euresys.Open_eVision_2_10.Easy3D.E3DTransformMatrix WorldToMapMatrix
{ get; }
```

EZMap8.WorldToZMap

Transforms a 3D world position to a 3D [EZMap8](#) position.
The ZMap space origin is at the lower left corner of the image.
The scales of the world space and ZMap space are the same.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void WorldToZMap(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint worldPt,
    ref Euresys.Open_eVision_2_10.Easy3D.E3DPoint zmapPt
)
```

Parameters

worldPt

Position in the 3D world space.

zmapPt

Position in the ZMap space.

Remarks

Do not use this method with the same variable as input and output. It might lead to incorrect results.

EZMap8.XResolution

Resolution of the [EZMap8](#) along the X axis
The resolution is the number of metric units per pixel.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
override float XResolution  
  
    { get; set; }
```

EZMap8.YResolution

Resolution of the [EZMap8](#) along the Y axis
The resolution is the number of metric units per pixel.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
override float YResolution  
  
    { get; set; }
```

EZMap8.ZMapToImage

Converts a 2D coordinate in the [EZMap8](#) space to image (sub)pixel space.
(x,y) is the ZMap position (which has the same scale as the world space).
(u,v) is the corresponding pixel position (with its origin in the upper left corner of the image).
All values are expressed in floating point numbers.
Returns TRUE if the pixel position is inside the image limits.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool ZMapToImage (
    float x,
    float y,
    ref float u,
    ref float v
)
```

Parameters

- x*
Position along horizontal axis in the ZMap space.
- y*
Position along vertical axis in the ZMap space.
- u*
Column of the pixel as a floating point value.
- v*
Row of the pixel as a floating point value.

EZMap8.ZMapToWorld

Transforms a 3D [EZMap8](#) position to a 3D world space position.
The ZMap space origin is at the lower left corner of the image.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void ZMapToWorld(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint zmapPt,
    ref Euresys.Open_eVision_2_10.Easy3D.E3DPoint worldPt
)
```

Parameters

zmapPt

Position in the ZMap space.

worldPt

Position in the 3D world space.

Remarks

Do not use this method with the same variable as input and output. It might lead to incorrect results.

EZMap8.ZResolution

Resolution of the [EZMap8](#) along the Z axis
The resolution is the number of grey values per pixel.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
override float ZResolution
{ get; set; }
```

4.198. EZMapToPointCloudConverter Class

Generates an [EPointCloud](#) from a ZMap.

Namespace: Euresys.Open_eVision_2_10.Easy3D

Methods

Convert

Generates an [EPointCloud](#) from a ZMap ([EZMap8](#), [EZMap16](#) or [EZMap32f](#)). ZMap pixel positions/values (U,V,W) are converted to 3D positions (X,Y,Z) and written to the given point cloud. By default, the target coordinate system is the original world space. Optional parameter `inZMapSpace` can switch to the ZMap space as the target coordinate system.

EZMapToPointCloudConverter

Creates an [EZMapToPointCloudConverter](#) object.

Z

MapToPointCloudConverter.Convert

Generates an [EPointCloud](#) from a ZMap ([EZMap8](#), [EZMap16](#) or [EZMap32f](#)). ZMap pixel positions/values (U,V,W) are converted to 3D positions (X,Y,Z) and written to the given point cloud. By default, the target coordinate system is the original world space. Optional parameter `inZMapSpace` can switch to the ZMap space as the target coordinate system.

Namespace: `Euresys.Open_eVision_2_10.Easy3D`

```
[C#]
void Convert(
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 srcZMap,
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud pointCloud,
    bool inZMapSpace
)

void Convert(
    Euresys.Open_eVision_2_10.Easy3D.EZMap8 srcZMap,
    Euresys.Open_eVision_2_10.ERegion region,
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud pointCloud,
    bool inZMapSpace
)
```

```

void Convert(
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 srcZMap,
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud pointCloud,
    bool inZMapSpace
)

void Convert(
    Euresys.Open_eVision_2_10.Easy3D.EZMap16 srcZMap,
    Euresys.Open_eVision_2_10.ERegion region,
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud pointCloud,
    bool inZMapSpace
)

void Convert(
    Euresys.Open_eVision_2_10.Easy3D.EZMap32f srcZMap,
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud pointCloud,
    bool inZMapSpace
)

void Convert(
    Euresys.Open_eVision_2_10.Easy3D.EZMap32f srcZMap,
    Euresys.Open_eVision_2_10.ERegion region,
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud pointCloud,
    bool inZMapSpace
)

void Convert(
    Euresys.Open_eVision_2_10.Easy3D.EZMap srcZMap,
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud pointCloud,
    bool inZMapSpace
)

void Convert(
    Euresys.Open_eVision_2_10.Easy3D.EZMap srcZMap,
    Euresys.Open_eVision_2_10.ERegion region,
    Euresys.Open_eVision_2_10.Easy3D.EPointCloud pointCloud,
    bool inZMapSpace
)

```

Parameters

srcZMap

The ZMap to convert.

pointCloud

The destination point cloud.

inZMapSpace

When TRUE, converts to 3D ZMap space instead of world space (default is FALSE).

region

The region of interest (default is no region).

Remarks

The destination point cloud will be cleared before being (re-)populated.

EZMapToPointCloudConverter.EZMapToPointCloudConverter

Creates an [EZMapToPointCloudConverter](#) object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void EZMapToPointCloudConverter (
)
void EZMapToPointCloudConverter (
    Euresys.Open_eVision_2_10.Easy3D.EZMapToPointCloudConverter other
)
```

Parameters

other

-

5. Structures

5.1. E3DPoint Struct

Represents a 3D point with floating point coordinates.

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

X	X coordinate of the point.
Y	Y coordinate of the point.
Z	Z coordinate of the point.

Methods

DistanceTo	Returns the euclidean distance to another E3DPoint .
E3DPoint	Constructs a default E3DPoint object.
operator!=	Checks if two E3DPoint are stricly different.
operator==	Checks if two E3DPoint are stricly equal.
Serialize	Serializes the E3DPoint .

E3DPoint.DistanceTo

Returns the euclidean distance to another [E3DPoint](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
float DistanceTo(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint point
)
```

Parameters

point

The other point.

E3DPoint.E3DPoint

Constructs a default [E3DPoint](#) object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void E3DPoint(
)

void E3DPoint(
    float x,
    float y,
    float z
)
```

Parameters

x

X coordinate of the point.

y

Y coordinate of the point.

z

Z coordinate of the point.

Remarks

If the default constructor is used, the point is initialized to (0, 0, 0).

E3DPoint.operator!=

Checks if two [E3DPoint](#) are strictly different.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool operator!=(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint point
)
```

Parameters

point

The other point.

E3DPoint.operator==

Checks if two [E3DPoint](#) are strictly equal.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
bool operator==(
    Euresys.Open_eVision_2_10.Easy3D.E3DPoint point
)
```

Parameters

point

The other point.

E3DPoint.Serialize

Serializes the [E3DPoint](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
void Serialize(
    Euresys.Open_eVision_2_10.ESerializer serializer
)
```

Parameters

serializer

The [ESerializer](#) object that is read from or written to.

E3DPoint.X

X coordinate of the point.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
float X
```

E3DPoint.Y

Y coordinate of the point.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
float Y
```

E3DPoint.Z

Z coordinate of the point.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```

```
float Z
```

5.2. EBW1 Struct

Black and white pixel value, coded as an unsigned 32-bit integer.

Remarks

Every pixel is coded on 1 bit, allowing to represent 2 different values. The value **0** stands for black (background), and the value **1** stands for white (foreground).

Namespace: Euresys.Open_eVision_2_10

Properties

Size	The number of bits in a pixel
Value	The value of the pixel.

Methods

EBW1	Constructs a EBW1 object.
------	---

EBW1.EBW1

Constructs a [EBW1](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EBW1 (
)
void EBW1 (
    uint value
)
```

Parameters

value

The value of the pixel.

EBW1.Size

The number of bits in a pixel

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
static int Size  
    { get; }
```

EBW1.Value

The value of the pixel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint Value
```

5.3. EBW16 Struct

Gray-level pixel value, coded as an unsigned 16-bit integer.

Remarks

High-quality cameras or scanners are able to digitize on 10 or 12 bits. Sometimes too, to avoid numerical truncation errors, intermediate processing results require more than 8 bits of storage. In such situations, 8 bits gray-level images are no longer sufficient. 16 bits gray-level images are similar to 8 bits ones, but each pixel is, in this case, coded on 16 bits, which effect is to increase the levels of gray to 65,536. It is not possible to show the difference between a gray-level image quantized on 16 bits rather than 8. Under Windows, no display device is able to display 16-bit gray levels. Windows doesn't allow you to display more than 256 gray levels.

Namespace: Euresys.Open_eVision_2_10

Properties

Size	The number of bits in a pixel
Value	The value of the pixel.

Methods

EBW16	Constructs a EBW16 object.
-------	--

EBW16.EBW16

Constructs a [EBW16](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EBW16 (
)
void EBW16 (
    ushort value
)
```

Parameters

value

The value of the pixel.

EBW16.Size

The number of bits in a pixel

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
static int Size  
  
    { get; }
```

EBW16.Value

The value of the pixel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
ushort Value
```

5.4. EBW16Path Struct

Path from a [EBW16](#) image: image pixel coordinates, and associated gray-level pixel value.

Namespace: Euresys.Open_eVision_2_10

Properties

Pixel	The value of the pixel at this coordinate.
X	Coordinate along the horizontal direction.
Y	Coordinate along the vertical direction.

EBW16Path.Pixel

The value of the pixel at this coordinate.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EBW16 Pixel
```

EBW16Path.X

Coordinate along the horizontal direction.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int x
```

EBW16Path.Y

Coordinate along the vertical direction.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int Y
```

5.5. EBW32 Struct

Gray-level pixel value, coded as an unsigned 32-bit integer.

Namespace: Euresys.Open_eVision_2_10

Properties

Size	The number of bits in a pixel
Value	The value of the pixel.

Methods

EBW32	Constructs a EBW32 object.
-----------------------	--

EBW32.EBW32

Constructs a [EBW32](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void EBW32 (  
    )  
void EBW32 (  
    uint value  
    )
```

Parameters

value

The value of the pixel.

EBW32.Size

The number of bits in a pixel

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
static int Size  
    { get; }
```

EBW32.Value

The value of the pixel.

Namespace: Euresys.Open_eVision_2_10

[C#]

uint Value

5.6. EBW8 Struct

Gray-level pixel value, coded as an unsigned 8-bit integer.

Remarks

Every pixel is coded on 8 bits, allowing to represent 256 different values. The value **0** stands for black (background) and the value **255** stands for white (foreground). The 254 remaining values stand for shades of gray. This is sufficient for most applications. Most of the Open eVision gray-level operations apply to this pixel type.

Namespace: Euresys.Open_eVision_2_10

Properties

Size	The number of bits in a pixel
Value	The value of the pixel.

Methods

EBW8	Constructs a EBW8 object.
------	---

EBW8.EBW8

Constructs a [EBW8](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EBW8 (
)
void EBW8 (
    byte value
)
```

Parameters

value

The value of the pixel.

EBW8.Size

The number of bits in a pixel

Namespace: Euresys.Open_eVision_2_10

```
[C#]
static int Size
{ get; }
```

EBW8.Value

The value of the pixel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

byte Value

5.7. EBW8Path Struct

Path from a [EBW8](#) image: image pixel coordinates, and associated gray-level pixel value.

Namespace: Euresys.Open_eVision_2_10

Properties

Pixel	The value of the pixel at this coordinate.
X	Coordinate along the horizontal direction.
Y	Coordinate along the vertical direction.

EBW8Path.Pixel

The value of the pixel at this coordinate.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
Euresys.Open_eVision_2_10.EBW8 Pixel
```

EBW8Path.X

Coordinate along the horizontal direction.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int X
```

EBW8Path.Y

Coordinate along the vertical direction.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int Y
```

5.8. EC15 Struct

Color pixel value, coded as 3 fields of 5 bits each (red, green, blue components) and 1 field of 1 bit for padding.

Remarks

This class is suited to handle the Windows RGB15 color images. The pixel values are coded on 15 bits, leaving 32 possible levels per color component (red, green or blue).

Namespace: Euresys.Open_eVision_2_10

Properties

C0	The value of the first component of the pixel (red channel).
C1	The value of the second component of the pixel (green channel).
C2	The value of the third component of the pixel (blue channel).
Size	The number of bits in a pixel

Methods

EC15	Constructs a EC15 object.
------	---

EC15.C0

The value of the first component of the pixel (red channel).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
ushort C0
```

EC15.C1

The value of the second component of the pixel (green channel).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
ushort C1
```

EC15.C2

The value of the third component of the pixel (blue channel).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
ushort C2
```

EC15.EC15

Constructs a [EC15](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void EC15(  
    )
```

```
void EC15(  
    byte c0,  
    byte c1,  
    byte c2  
    )
```

Parameters

c0

The value of the first component of the pixel (red channel).

c1

The value of the second component of the pixel (green channel).

c2

The value of the third component of the pixel (blue channel).

EC15.Size

The number of bits in a pixel

Namespace: Euresys.Open_eVision_2_10

```
[C#]
static int Size
{ get; }
```

5.9. EC16 Struct

Color pixel value, coded as 3 fields of 5 bits, 6 bits and 5 bits (red, green and blue components).

Remarks

This class is suited to handle the Windows RGB16 color images. The pixel values are coded on 16 bits (5-6-5), leaving 32 possible levels for R and B components, and 64 possible levels for G component.

Namespace: Euresys.Open_eVision_2_10

Properties

C0	The value of the first component of the pixel (red channel).

C1	The value of the second component of the pixel (green channel).
C2	The value of the third component of the pixel (blue channel).
Size	The number of bits in a pixel

Methods

EC16	Constructs a EC16 object.
----------------------	---

EC16.C0

The value of the first component of the pixel (red channel).

Namespace: Euresys.Open_eVision_2_10

<pre>[C#] ushort C0</pre>

EC16.C1

The value of the second component of the pixel (green channel).

Namespace: Euresys.Open_eVision_2_10

<pre>[C#] ushort C1</pre>

EC16.C2

The value of the third component of the pixel (blue channel).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
ushort C2
```

EC16.EC16

Constructs a [EC16](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void EC16(  
)  
void EC16(  
    byte c0,  
    byte c1,  
    byte c2  
)
```

Parameters

c0

The value of the first component of the pixel (red channel).

c1

The value of the second component of the pixel (green channel).

c2

The value of the third component of the pixel (blue channel).

EC16.Size

The number of bits in a pixel

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
static int Size  
{ get; }
```

5.10. EC24 Struct

Color pixel value coded as 3 unsigned 8-bit integers (red, green and blue components).

Remarks

(RGB triplet, windows 24 bpp bitmap format) The pixel values are coded on 24 bits, providing 256 possible levels per color component. This way, RGB images can represent 16,777,216 different colors. This is sufficient for most applications. Most of the Open eVision color operations apply to this pixel type.

Namespace: Euresys.Open_eVision_2_10

Properties

C0	The value of the first component of the pixel (red channel).
C1	The value of the second component of the pixel (green channel).
C2	The value of the third component of the pixel (blue channel).

Size

M The number of bits in a pixel

e

thods

EC24

Constructs a **EC24** object.

EC24.C0

The value of the first component of the pixel (red channel).

Namespace: Euresys.Open_eVision_2_10

[C#]

byte C0

EC24.C1

The value of the second component of the pixel (green channel).

Namespace: Euresys.Open_eVision_2_10

[C#]

byte C1

EC24.C2

The value of the third component of the pixel (blue channel).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
byte C2
```

EC24.EC24

Constructs a [EC24](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void EC24(  
)  
void EC24(  
    Euresys.Open_eVision_2_10.ERGBColor rgbColor  
)  
void EC24(  
    byte c0,  
    byte c1,  
    byte c2  
)
```

Parameters

rgbColor

-

c0

The value of the first component of the pixel (red channel).

c1

The value of the second component of the pixel (green channel).

c2

The value of the third component of the pixel (blue channel).

EC24.Size

The number of bits in a pixel

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
static int Size  
    { get; }
```

5.11. EC24A Struct

Color pixel value coded as 4 unsigned 8-bit integers (red, green, blue and alpha components).

Remarks

This class is suited to handle the Windows RGB32 color format. The pixel values are coded on 32 bits, leaving 256 possible levels per color component (red, green or blue), and 8 more bits for an alpha channel. Currently, the alpha channel is not used for any purpose in the Open eVision processing functions. Users are free to use it to store an additional gray-level content.

Namespace: Euresys.Open_eVision_2_10

Properties

A	The value of the alpha component of the pixel.
---	--

C0	The value of the first component of the pixel (red channel).
C1	The value of the second component of the pixel (green channel).
C2	The value of the third component of the pixel (blue channel).
Size	The number of bits in a pixel

Methods

EC24A	Constructs a EC24A object.
-------	--

EC24A.A

The value of the alpha component of the pixel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
byte A
```

EC24A.C0

The value of the first component of the pixel (red channel).

Namespace: Euresys.Open_eVision_2_10

[C#]

byte C0

EC24A.C1

The value of the second component of the pixel (green channel).

Namespace: Euresys.Open_eVision_2_10

[C#]

byte C1

EC24A.C2

The value of the third component of the pixel (blue channel).

Namespace: Euresys.Open_eVision_2_10

[C#]

byte C2

EC24A.EC24A

Constructs a [EC24A](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EC24A(
)

void EC24A(
    byte c0,
    byte c1,
    byte c2,
    byte a
)
```

Parameters

c0

The value of the first component of the pixel (red channel).

c1

The value of the second component of the pixel (green channel).

c2

The value of the third component of the pixel (blue channel).

a

-

EC24A.Size

The number of bits in a pixel

Namespace: Euresys.Open_eVision_2_10

```
[C#]
static int Size
{ get; }
```

5.12. EC24Path Struct

Path from a [EC24](#) image: image pixel coordinates, and associated color pixel value.

Namespace: Euresys.Open_eVision_2_10

Properties

Pixel	The value of the pixel at this coordinate.
X	Coordinate along the horizontal direction.
Y	Coordinate along the vertical direction.

EC24Path.Pixel

The value of the pixel at this coordinate.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EC24 Pixel
```

EC24Path.X

Coordinate along the horizontal direction.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
int x
```

EC24Path.Y

Coordinate along the vertical direction.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
int y
```

5.13. EC48 Struct

Color pixel value coded as 3 unsigned 16-bit integers (red, green, blue components).

Namespace: Euresys.Open_eVision_2_10

Properties

C0	The value of the first component of the pixel (red channel).
C1	The value of the second component of the pixel (green channel).
C2	The value of the third component of the pixel (blue channel).
Size	The size of a pixel.

Methods

EC48

Constructs a [EC48](#) object.

EC48.C0

The value of the first component of the pixel (red channel).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
ushort C0
```

EC48.C1

The value of the second component of the pixel (green channel).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
ushort C1
```

EC48.C2

The value of the third component of the pixel (blue channel).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
ushort c2
```

EC48.EC48

Constructs a [EC48](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void EC48(  
    )  
  
void EC48(  
    ushort c0,  
    ushort c1,  
    ushort c2  
    )
```

Parameters

c0

The value of the first component of the pixel (red channel).

c1

The value of the second component of the pixel (green channel).

c2

The value of the third component of the pixel (blue channel).

EC48.Size

The size of a pixel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
static int Size  
    { get; }
```

5.14. EColor Struct

Triple of floating-point numbers that encode a color in a given color system.

Namespace: Euresys.Open_eVision_2_10

Properties

C0	First color component.
C1	Second color component.
C2	Third color component.

Methods

EColor	Constructor for EColor objects.
--------	---------------------------------

EColor.C0

First color component.

Namespace: Euresys.Open_eVision_2_10

[C#]

float C0

EColor.C1

Second color component.

Namespace: Euresys.Open_eVision_2_10

[C#]

float C1

EColor.C2

Third color component.

Namespace: Euresys.Open_eVision_2_10

[C#]

float C2

EColor.EColor

Constructor for [EColor](#) objects.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
void EColor(
)
void EColor(
float c0,
float c1,
float c2
)
```

Parameters

- c0*
value for the first color component
- c1*
value for the second color component
- c2*
value for the third color component

5.15. EDepth16 Struct

Depth value of the pixel, coded as an unsigned 16-bit integer.

Namespace: Euresys.Open_eVision_2_10

Properties

Size	The number of bits in a pixel
Value	The depth value of the pixel.

Methods

EDepth16	Constructs a EDepth16 object.
----------	---

EDepth16.EDepth16

Constructs a [EDepth16](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
void EDepth16(  
    )  
  
void EDepth16(  
    ushort value  
    )
```

Parameters

value

The depth value of the pixel.

EDepth16.Size

The number of bits in a pixel

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
static int Size  
  
    { get; }
```

EDepth16.Value

The depth value of the pixel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
ushort Value
```

5.16. EDepth32f Struct

Depth value of the pixel, coded as a 32-bits floating point value.

Namespace: Euresys.Open_eVision_2_10

Properties

Size	The number of bits in a pixel
Value	The depth value of the pixel.

Methods

EDepth32f	Constructs a EDepth32f object.
---------------------------	--

EDepth32f.EDepth32f

Constructs a [EDepth32f](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void EDepth32f(  
    )  
void EDepth32f(  
    float value  
    )
```

Parameters

value

The depth value of the pixel.

EDepth32f.Size

The number of bits in a pixel

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
static int Size  
    { get; }
```

EDepth32f.Value

The depth value of the pixel.

Namespace: Euresys.Open_eVision_2_10

[C#]

`float Value`

5.17. EDepth8 Struct

Depth value of the pixel, coded as an unsigned 8-bit integer.

Namespace: Euresys.Open_eVision_2_10

Properties

Size	The number of bits in a pixel
Value	The depth value of the pixel.

Methods

EDepth8	Constructs a EDepth8 object.
---------	--

EDepth8.EDepth8

Constructs a [EDepth8](#) object.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void EDepth8(  
    )  
void EDepth8(  
    byte value  
    )
```

Parameters

value

The depth value of the pixel.

EDepth8.Size

The number of bits in a pixel

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
static int Size  
    { get; }
```

EDepth8.Value

The depth value of the pixel.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
byte Value
```


5.18. EFeatureData Struct

Describes object features.

Remarks

A feature is associated to an array of values, each corresponding to an object of given identification number. A feature is also characterized by the size of the array, a feature number, data size/type information and pointers to both ends of the array. The features can be accessed by their number (see [EFeature](#)). To obtain the value of a given feature of a given object, just use the class member [ECodedImage::GetObjectFeature](#). This enumeration pertains to the EasyObject legacy API and should not be used for new developments. See [ECodedImage2](#) for the new API.

Namespace: Euresys.Open_eVision_2_10

Properties

FeatDataSize	Data size (see EDataSize).
FeatDataType	Data type (see EDataType).
FeatNum	Code (see EFeature).
Size	Number of objects for which the feature is stored.

EFeatureData.FeatDataSize

Data size (see [EDataSize](#)).

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
Euresys.Open_eVision_2_10.EDataSize FeatDataSize
```

EFeatureData.FeatDataType

Data type (see [EDataType](#)).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.EDataType FeatDataType
```

EFeatureData.FeatNum

Code (see [EFeature](#)).

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
Euresys.Open_eVision_2_10.ELegacyFeature FeatNum
```

EFeatureData.Size

Number of objects for which the feature is stored.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
int Size
```

5.19. EISH Struct

Intensity, Saturation, Hue color system.

Namespace: Euresys.Open_eVision_2_10

Properties

H	Hue component.
I	Intensity component.
S	Saturation component.

EISH.H

Hue component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float H
```

EISH.I

Intensity component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float I
```

EISH.S

Saturation component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float S
```

5.20. ELAB Struct

CIE Lightness, a*, b* color system.

Namespace: Euresys.Open_eVision_2_10

Properties

A	a* component.
---	---------------

B	b* component.
L	Lightness component.

ELAB.A

a* component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float A
```

ELAB.B

b* component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float B
```

ELAB.L

Lightness component.

Namespace: Euresys.Open_eVision_2_10

[C#]

float L

5.21. ELCH Struct

Lightness, Chroma, Hue color system.

Namespace: Euresys.Open_eVision_2_10

Properties

C	Chroma component.
H	Hue component.
L	Lightness component.

ELCH.C

Chroma component.

Namespace: Euresys.Open_eVision_2_10

[C#]

float C

ELCH.H

Hue component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float H
```

ELCH.L

Lightness component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float L
```

5.22. ELSH Struct

Lightness, Saturation, Hue color system.

Namespace: Euresys.Open_eVision_2_10

Properties

H	Hue component.
---	----------------

L	Lightness component.
S	Saturation component.

ELSH.H

Hue component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float H
```

ELSH.L

Lightness component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float L
```

ELSH.S

Saturation component.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
```

```
float S
```

5.23. ELUV Struct

CIE Lightness, u^* , v^* color system.

Namespace: Euresys.Open_eVision_2_10

Properties

L	Lightness component.
U	u^* component.
V	v^* component.

ELUV.L

Lightness component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float L
```

ELUV.U

u* component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float U
```

ELUV.V

v* component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float V
```

5.24. EMatchPosition Struct

Represents a single instance of the pattern in the search field, as returned by the EasyMatch matching process.

Remarks

[EMatcher::GetPosition](#) returns instances of this class. A [EMatchPosition](#) object represents one matched instance, with all the needed information about it.

Namespace: Euresys.Open_eVision_2_10

Properties

Angle	Clockwise rotation angle, expressed using the current angle unit, of the pattern found in the image.
AreaRatio	Ratio between the found pattern area inside the search ROI and its complete area.
CenterX	Abscissa of the center of the pattern found in the image.
CenterY	Ordinate of the center of the pattern found in the image.
Interpolated	Indicates whether sub-pixel interpolation was actually performed on the position.
Scale	Scale factor of the pattern found in the image.
ScaleX	Measured horizontal scaling of the pattern found in the image, expressed as a dimensionless ratio.
ScaleY	Measured vertical scaling of the pattern found in the image, expressed as a dimensionless ratio.
Score	Indicates how good a matching was.

EMatchPosition.Angle

Clockwise rotation angle, expressed using the current angle unit, of the pattern found in the image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float Angle
```

Remarks

0 if no rotation is allowed.

EMatchPosition.AreaRatio

Ratio between the found pattern area inside the search ROI and its complete area.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float AreaRatio
```

EMatchPosition.CenterX

Abscissa of the center of the pattern found in the image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float CenterX
```

EMatchPosition.CenterY

Ordinate of the center of the pattern found in the image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float CenterY
```

EMatchPosition.Interpolated

Indicates whether sub-pixel interpolation was actually performed on the position.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool Interpolated
```

Remarks

In some cases, when the pattern is found close to the ROI edge, sub-pixel interpolation cannot be used.

EMatchPosition.Scale

Scale factor of the pattern found in the image.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float Scale
```

Remarks

1 if no scaling is allowed.

EMatchPosition.ScaleX

Measured horizontal scaling of the pattern found in the image, expressed as a dimensionless ratio.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ScaleX
```

EMatchPosition.ScaleY

Measured vertical scaling of the pattern found in the image, expressed as a dimensionless ratio.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float ScaleY
```

EMatchPosition.Score

Indicates how good a matching was.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Score
```

Remarks

1 means that the matching was perfect. Lower values correspond to approximate matching; **-1** corresponds to a perfect mismatch (pattern superimposed on its negative image).

5.25. EMatrixCodeIso15415GradingParameters Struct

Holds all grading parameters pertaining to ISO/IEC 15415

Namespace: Euresys.Open_eVision_2_10

Properties

AxialNonUniformity	Axial Non Uniformity
AxialNonUniformityGrade	Axial Non Uniformity Grade
DecodingGrade	Decoding Grade
FixedPatternDamageGrade	Fixed Pattern Damage Grade

GridNonUniformity	Grid Non Uniformity
GridNonUniformityGrade	Grid Non Uniformity Grade
HorizontalPrintGrowth	Horizontal Print Growth
ModulationGrade	Modulation Grade
OverallSymbolGrade	Overall Symbol Grade
ReflectanceMarginGrade	Reflectance Margin Grade
SymbolContrast	Symbol Contrast
SymbolContrastGrade	Symbol Contrast Grade
UnusedErrorCorrection	Unused Error Correction
UnusedErrorCorrectionGrade	Unused Error Correction Grade
VerticalPrintGrowth	Vertical Print Growth

Methods

EMatrixCodeIso15415GradingParameters	-
--------------------------------------	---

EMatrixCodeIso15415GradingParameters.AxialNonUniformity

Axial Non Uniformity

Namespace: Euresys.Open_eVision_2_10


```
[C#]
```

```
float AxialNonUniformity
```

EMatrixCodeIso15415GradingParameters.AxialNonUniformityGrade

Axial Non Uniformity Grade

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
int AxialNonUniformityGrade
```

EMatrixCodeIso15415GradingParameters.DecodingGrade

Decoding Grade

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
int DecodingGrade
```

EMatrixCodeIso15415GradingParameters.EMatrixCodeIso15415GradingParameters

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void EMatrixCodeIso15415GradingParameters (  
)
```

EMatrixCodeIso15415GradingParameters.FixedPatternDamageGrade

Fixed Pattern Damage Grade

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int FixedPatternDamageGrade
```

EMatrixCodeIso15415GradingParameters.GridNonUniformity

Grid Non Uniformity

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float GridNonUniformity
```

EMatrixCodeIso15415GradingParameters.GridNonUniformityGrade

Grid Non Uniformity Grade

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
int GridNonUniformityGrade
```

EMatrixCodeIso15415GradingParameters.HorizontalPrintGrowth

Horizontal Print Growth

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float HorizontalPrintGrowth
```

EMatrixCodeIso15415GradingParameters.ModulationGrade

Modulation Grade

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int ModulationGrade
```

EMatrixCodeIso15415GradingParameters.OverallSymbolGrade

Overall Symbol Grade

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int OverallSymbolGrade
```

EMatrixCodeIso15415GradingParameters.ReflectanceMarginGrade

Reflectance Margin Grade

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int ReflectanceMarginGrade
```

EMatrixCodeIso15415GradingParameters.SymbolContrast

Symbol Contrast

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float SymbolContrast
```

EMatrixCodeIso15415GradingParameters.SymbolContrastGrade

Symbol Contrast Grade

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int SymbolContrastGrade
```

EMatrixCodeIso15415GradingParameters.UnusedErrorCorrection

Unused Error Correction

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float UnusedErrorCorrection
```

EMatrixCodeIso15415GradingParameters.UnusedErrorCorrection
Grade

Unused Error Correction Grade

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int UnusedErrorCorrectionGrade
```

EMatrixCodeIso15415GradingParameters.VerticalPrintGrowth

Vertical Print Growth

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float VerticalPrintGrowth
```

5.26. EMatrixCodeIso29158CalibrationParameters Struct

Holds all grading inputs pertaining to ISO/IEC 29158

Namespace: Euresys.Open_eVision_2_10

Properties

MLcal	Mean of the light from a histogram of the calibrated standard.
Rcal	Reported reflectance value, from a calibration standard.
SRcal	System response parameters(such as exposure and/or again) used to create an image of the calibration standard.
SRtarget	System response parameters(such as exposure and/or again) used to create an image of the symbole under test.

Methods

EMatrixCodeIso29158CalibrationParameters	-
--	---

EMatrixCodeIso29158CalibrationParameters.EMatrixCodeIso29158CalibrationParameters

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void EMatrixCodeIso29158CalibrationParameters (  
)
```

EMatrixCodeIso29158CalibrationParameters.MLcal

Mean of the light from a histogram of the calibrated standard.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float MLcal
```

EMatrixCodeIso29158CalibrationParameters.Rcal

Reported reflectance value, from a calibration standard.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Rcal
```

EMatrixCodeIso29158CalibrationParameters.SRcal

System response parameters(such as exposure and/or gain) used to create an image of the calibration standard.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
float SRcal
```

EMatrixCodeIso29158CalibrationParameters.SRtarget

System response parameters(such as exposure and/or again) used to create an image of the symbole under test.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
float SRtarget
```

5.27. EMatrixCodeIso29158GradingParameters Struct

Holds all grading parameters pertaining to ISO/IEC 29158

Namespace: Euresys.Open_eVision_2_10

Properties

CellContrastGrade	Cell Contrast Grade
CellModulationGrade	Cell Modulation Grade
FixedPatternDamageGrade	FixedPatternDamageGrade

IsMeanLightInRequiredBounds	Is Mean Light In Required Bounds
MeanLight	Mean Light
MinimumReflectanceGrade	Minimum Reflectance Grade
OverallSymbolGrade	Overall Symbol Grade

Methods

EMatrixCodeIso29158GradingParameters	-
--	---

EMatrixCodeIso29158GradingParameters.CellContrastGrade

Cell Contrast Grade

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int CellContrastGrade
```

EMatrixCodeIso29158GradingParameters.CellModulationGrade

Cell Modulation Grade

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
int CellModulationGrade
```

EMatrixCodeIso29158GradingParameters.EMatrixCodeIso29158GradingParameters

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void EMatrixCodeIso29158GradingParameters(  
)
```

EMatrixCodeIso29158GradingParameters.FixedPatternDamageGrade

FixedPatternDamageGrade

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
int FixedPatternDamageGrade
```

EMatrixCodeIso29158GradingParameters.IsMeanLightInRequiredBounds

Is Mean Light In Required Bounds

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool IsMeanLightInRequiredBounds
```

EMatrixCodeIso29158GradingParameters.MeanLight

Mean Light

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float MeanLight
```

EMatrixCodeIso29158GradingParameters.MinimumReflectanceGrade

Minimum Reflectance Grade

Namespace: Euresys.Open_eVision_2_10

[C#]

```
int MinimumReflectanceGrade
```

EMatrixCodeIso29158GradingParameters.OverallSymbolGrade

Overall Symbol Grade

Namespace: Euresys.Open_eVision_2_10

[C#]

```
int OverallSymbolGrade
```

5.28. EMatrixCodeSemiT10GradingParameters Struct

Holds all grading parameters pertaining to Semi T10-

Namespace: Euresys.Open_eVision_2_10

Properties

CellDefects	Cell Defects
DataMatrixCellHeight	Data Matrix Cell Height
DataMatrixCellWidth	Data Matrix Cell Width

FinderPatternDefects	Finder Pattern Defects
HorizontalMarkGrowth	Horizontal Mark Growth
HorizontalMarkMisplacement	Horizontal Mark Misplacement
SymbolContrast	Symbol Contrast
SymbolContrastSNR	Symbol Contrast SNR
UnusedErrorCorrection	Unused Error Correction
VerticalMarkGrowth	Vertical Mark Growth
VerticalMarkMisplacement	Vertical Mark Misplacement

Methods

EMatrixCodeSemiT10GradingParameters	-
-------------------------------------	---

EMatrixCodeSemiT10GradingParameters.CellDefects

Cell Defects

Namespace: Euresys.Open_eVision_2_10

```
[C#]
float CellDefects
```

EMatrixCodeSemiT10GradingParameters.DataMatrixCellHeight

Data Matrix Cell Height

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float DataMatrixCellHeight
```

EMatrixCodeSemiT10GradingParameters.DataMatrixCellWidth

Data Matrix Cell Width

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float DataMatrixCellWidth
```

EMatrixCodeSemiT10GradingParameters.EMatrixCodeSemiT10GradingParameters

-

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void EMatrixCodeSemiT10GradingParameters (  
)
```

EMatrixCodeSemiT10GradingParameters.FinderPatternDefects

Finder Pattern Defects

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float FinderPatternDefects
```

EMatrixCodeSemiT10GradingParameters.HorizontalMarkGrowth

Horizontal Mark Growth

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float HorizontalMarkGrowth
```

EMatrixCodeSemiT10GradingParameters.HorizontalMarkMisplacement

Horizontal Mark Misplacement

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float HorizontalMarkMisplacement
```

EMatrixCodeSemiT10GradingParameters.SymbolContrast

Symbol Contrast

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float SymbolContrast
```

EMatrixCodeSemiT10GradingParameters.SymbolContrastSNR

Symbol Contrast SNR

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float SymbolContrastSNR
```

EMatrixCodeSemiT10GradingParameters.UnusedErrorCorrection

Unused Error Correction

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float UnusedErrorCorrection
```

EMatrixCodeSemiT10GradingParameters.VerticalMarkGrowth

Vertical Mark Growth

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float VerticalMarkGrowth
```

EMatrixCodeSemiT10GradingParameters.VerticalMarkMisplacement

Vertical Mark Misplacement

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float VerticalMarkMisplacement
```

5.29. EMatrixPosition Struct

Represents a position in a 2D Matrix.

Namespace: Euresys.Open_eVision_2_10

Properties

X	X position.
Y	Y position.

Methods

EMatrixPosition	Constructs a default EMatrixPosition object.
<code>operator!=</code>	Checks if two EMatrixPosition are stricly different
<code>operator==</code>	Checks if two EMatrixPosition are stricly equal

EMatrixPosition.EMatrixPosition

Constructs a default [EMatrixPosition](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
void EMatrixPosition(  
)
```

```
void EMatrixPosition(  
    int x,  
    int y  
)
```

Parameters

x
X position.

y
Y position.

Remarks

If the default constructor is used, the position is initialized to (0, 0).

EMatrixPosition.operator!=

Checks if two [EMatrixPosition](#) are strictly different

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
bool operator!=(  
    Euresys.Open_eVision_2_10.EMatrixPosition position  
)
```

Parameters

position
The other position.

EMatrixPosition.operator==

Checks if two [EMatrixPosition](#) are strictly equal

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
bool operator==(  
    Euresys.Open_eVision_2_10.EMatrixPosition position  
)
```

Parameters

position

The other position.

EMatrixPosition.X

X position.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
int X
```

EMatrixPosition.Y

Y position.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
  
int Y
```

5.30. EObjectData Struct

Describes objects.

Remarks

An object is characterized by a class, a unique identification number, the number of its constituent runs, the number of its holes (if the object is a real object, not a hole), a selection flag, an identification flag (real object or hole) and the list of its constituent runs. After the object construction phase (real objects and eventually holes), all the objects are gathered in a single dynamic list. The objects can be accessed by their absolute position in the list as well as by their identification number. This structure pertains to the EasyObject legacy API and should not be used for new developments.

Note. After a sorting operation, the objects retain their identification number, not their absolute position in the list. If need be, the list of runs of an object can be traversed by means of the following functions: **GetObjNbRun**, [ECodedImage::GetObjFirstRunPtr](#), [ECodedImage::GetObjLastRunPtr](#).

Namespace: Euresys.Open_eVision_2_10

Properties

Class	Class code.
IsHole	TRUE if the object is a hole, FALSE otherwise.
IsSelected	Selection flag.
ObjNbHole	Number of holes.
ObjNbRun	Number of runs.
ObjNum	Identification number.

EObjectData.Class

Class code.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int Class
```

EObjectData.IsHole

TRUE if the object is a hole, **FALSE** otherwise.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
bool IsHole
```

EObjectData.IsSelected

Selection flag.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
byte IsSelected
```

EObjectData.ObjNbHole

Number of holes.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int ObjNbHole
```

EObjectData.ObjNbRun

Number of runs.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int ObjNbRun
```

EObjectData.ObjNum

Identification number.

Namespace: Euresys.Open_eVision_2_10


```
[C#]
```

```
int ObjNum
```

5.31. EOOCR2CharacterCandidate Struct

Holds a single recognition score for a detected character from the image

Remarks

The variable "code" contains the ASCII-representation of the reference character from the database. The variable "score" contains the recognition score between the detected character and the reference character.

Namespace: Euresys.Open_eVision_2_10

Properties

Code	Contains the ASCII-representation of the reference character from the database.
Score	Contains the recognition score between the detected character and the reference character.

Methods

EOOCR2CharacterCandidate	Constructs an EOOCR2CharacterCandidate context.
--	---

EOOCR2CharacterCandidate.Code

Contains the ASCII-representation of the reference character from the database.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
ushort Code
```

EOCR2CharacterCandidate.EOCR2CharacterCandidate

Constructs an [EOCR2CharacterCandidate](#) context.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void EOCR2CharacterCandidate (  
    )  
  
void EOCR2CharacterCandidate (  
    ushort code,  
    float score  
    )
```

Parameters

code

-

score

-

EOCR2CharacterCandidate.Score

Contains the recognition score between the detected character and the reference character.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float Score
```

5.32. EPath Struct

Path from an image: image pixel coordinates.

Namespace: Euresys.Open_eVision_2_10

Properties

X	Coordinate along the horizontal direction.
Y	Coordinate along the vertical direction.

EPath.X

Coordinate along the horizontal direction.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
int X
```

EPath.Y

Coordinate along the vertical direction.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
int Y
```

5.33. EPeak Struct

Represents a peak in a profile.

Namespace: Euresys.Open_eVision_2_10

Properties

Amplitude	Amplitude of the peak.
Area	Area of the peak.
Center	Center of the peak.
Length	Length of the peak.
Start	Start of the peak.

EPeak.Amplitude

Amplitude of the peak.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int Amplitude
```

EPeak.Area

Area of the peak.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int Area
```

EPeak.Center

Center of the peak.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float Center
```

EPeak.Length

Length of the peak.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint Length
```

EPeak.Start

Start of the peak.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
uint Start
```

5.34. ERenderStyle Struct

Represents how to visualize 3D Objects in a E3DViewer

Namespace: Euresys.Open_eVision_2_10.Easy3D

Properties

fillRGB	Fill (inside) color of the object
hasFill	Indicates that the inside of the object should be rendered.
hasLine	Indicates that the edges of the object should be rendered
lineRGB	Line (edge) color of the object.
pointRGB	Color of the object if it is a point.

Methods

ERenderStyle	Constructs a default ERenderStyle object.
------------------------------	---

ERenderStyle.ERenderStyle

Constructs a default [ERenderStyle](#) object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
void ERenderStyle(  
)
```

ERenderStyle.fillRGB

Fill (inside) color of the object

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.EC24A fillRGB
```

Remarks

Only applied if the object is a polygon.

ERenderStyle.hasFill

Indicates that the inside of the object should be rendered.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
bool hasFill
```

Remarks

Only applied if the object is a polygon.

ERenderStyle.hasLine

Indicates that the edges of the object should be rendered

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]
```



```
bool hasLine
```

Remarks

Only applied if the object is a polygon.

ERenderStyle.lineRGB

Line (edge) color of the object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.EC24A lineRGB
```

Remarks

Only applied if the object is a polygon.

ERenderStyle.pointRGB

Color of the object if it is a point.

Namespace: Euresys.Open_eVision_2_10.Easy3D

```
[C#]  
Euresys.Open_eVision_2_10.EC24A pointRGB
```

5.35. ERGB Struct

NTSC/PAL/SMPTE Red, Green, Blue color system.

Namespace: Euresys.Open_eVision_2_10

Properties

B	Blue component.
G	Green component.
R	Red component.

ERGB.B

Blue component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float B
```

ERGB.G

Green component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float G
```

ERGB.R

Red component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float R
```

5.36. ERGBColor Struct

NTSC/PAL/SMPTE Red, Green, Blue color system.

Namespace: Euresys.Open_eVision_2_10

Properties

Blue	Blue component.
Green	Green component.
Red	Red component.

Methods

--

[ERGBColor](#)

Constructs an [ERGBColor](#) object.

E

R

GBColor.Blue

Blue component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
int Blue
```

ERGBColor.ERGBColor

Constructs an [ERGBColor](#) object.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
void ERGBColor(  
    int red,  
    int green,  
    int blue  
)  
  
void ERGBColor(  
)
```

Parameters

red

Red component.

green

Green component.

blue

Blue component.

ERGBColor.Green

Green component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
int Green
```

ERGBColor.Red

Red component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
int Red
```

5.37. ERun Struct

Namespace: Euresys.Open_eVision_2_10

Properties

Length	Length of the Run
OrgX	X origin of the Run
Y	Y position of the Run

Methods

ERun	Constructs an ERun context.
operator!=	Checks if this instance is not strictly equal to another
operator==	Checks if this instance is strictly equal to another

ERun.ERun

Constructs an ERun context.

Namespace: Euresys.Open_eVision_2_10

[C#]

```
void ERun (
)

void ERun (
    int orgX,
    int length,
    int y
)
```

Parameters

orgX

X origin of the Run.

length

Length of the Run.

y

Y position of the Run.

ERun.Length

Length of the Run

Namespace: Euresys.Open_eVision_2_10

[C#]

```
int Length
```

ERun.operator!=

Checks if this instance is not strictly equal to another

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool operator!=(
    Euresys.Open_eVision_2_10.ERun other
)
```

Parameters

other

Reference to the other [ERun](#) instance

ERun.operator!=

Checks if this instance is strictly equal to another

Namespace: Euresys.Open_eVision_2_10

```
[C#]
bool operator==(
    Euresys.Open_eVision_2_10.ERun other
)
```

Parameters

other

Reference to the other [ERun](#) instance

ERun.OrgX

X origin of the Run

Namespace: Euresys.Open_eVision_2_10


```
[C#]
```

```
int OrgX
```

ERun.Y

Y position of the Run

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
int Y
```

5.38. ERunData Struct

Describes runs.

Remarks

A run is characterized by a starting point (**OrgX**, **OrgY**), by a length, a class, a unique identification number and the number of the object to which they belong. After the run construction phase, all the runs are gathered in a single dynamic list.

Namespace: Euresys.Open_eVision_2_10

Properties

Class	Class.
Len	Length.

ObjNum	Identification number of the object to which the run belongs.
OrgX	Start point abscissa.
OrgY	Start point ordinate.

ERunData.Class

Class.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int Class
```

ERunData.Len

Length.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int Len
```

ERunData.ObjNum

Identification number of the object to which the run belongs.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int ObjNum
```

ERunData.OrgX

Start point abscissa.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int OrgX
```

ERunData.OrgY

Start point ordinate.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
int OrgY
```

5.39. EVSH Struct

Value, Saturation, Hue color system.

Namespace: Euresys.Open_eVision_2_10

Properties

H	Hue component.
S	Saturation component.
V	Value component.

EVSH.H

Hue component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float H
```

EVSH.S

Saturation component.

Namespace: Euresys.Open_eVision_2_10

[C#]

float s

EVSH.V

Value component.

Namespace: Euresys.Open_eVision_2_10

[C#]

float v

5.40. EXYZ Struct

CIE XYZ color system.

Namespace: Euresys.Open_eVision_2_10

Properties

X	X component.
Y	Y component.
Z	Z component.

EXYZ.X

X component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float X
```

EXYZ.Y

Y component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Y
```

EXYZ.Z

Z component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float Z
```

5.41. EYIQ Struct

CCIR Luma, Inphase, Quadrature color system.

Namespace: Euresys.Open_eVision_2_10

Properties

I	Inphase component.
Q	Quadrature component.
Y	Luma component.

EYIQ.I

Inphase component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float I
```

EYIQ.Q

Quadrature component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Q
```

EYIQ.Y

Luma component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Y
```

5.42. EYSH Struct

CCIR Luma, Saturation, Hue color system.

Namespace: Euresys.Open_eVision_2_10

Properties

H	Hue component.
---	----------------

S	Saturation component.
Y	Luma component.

EYSH.H

Hue component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float H
```

EYSH.S

Saturation component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float S
```

EYSH.Y

Luma component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float Y
```

5.43. EYUV Struct

CCIR Luma, U Chroma, V Chroma color system.

Namespace: Euresys.Open_eVision_2_10

Properties

U	U Chroma component.
V	V Chroma component.
Y	Luma component.

EYUV.U

U Chroma component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]
```

```
float U
```

EYUV.V

V Chroma component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float V
```

EYUV.Y

Luma component.

Namespace: Euresys.Open_eVision_2_10

```
[C#]  
float Y
```

6. Enumerations

6.1. E3DObjectFeature Enum

The list of possible extracted features for [E3DObject](#)

Namespace: Euresys.Open_eVision_2_10.Easy3D

Length	Length of the object in metric units.
Width	Width of the object in metric units.
LocalHeight	Local height of the object in metric units.
ReferenceHeight	Reference height of the object in metric units.
Orientation	Orientation of the object.
BoundingBox	3D oriented bounding box of the object.
AveragePosition	3D average position of the object.
LocalTopPosition	3D highest position of the object relatively to the object base plane.
ReferenceTopPosition	3D top position relative to the ZMap origin (this is the position with the highest Z coordinate).
Plane	Plane fitted to the object 3D positions.

LocalTilt	Angle between the object plane and the base plane.
ReferenceTilt	Angle between the object plane and the vertical (Z) axis.
Area	Object area in metric units.
Volume	Object volume in metric units.
BasePlane	Base plane for the object.
BaseTilt	Angle between the base plane and the vertical (Z) axis.
ERectangleRegion	ERectangleRegion enclosing the object ZMap pixels.
ERegion	ERegion composed of the object ZMap pixels.

6.2. EAdaptiveThresholdMethod Enum

Adaptive thresholding modes.

Namespace: Euresys.Open_eVision_2_10

Mean	Use the mean as threshold.
Median	Use the median as threshold.
Middle	Use the middle of the values interval as threshold.

6.3. EAlignmentPolarity Enum

Polarity of an alignment, used in [EFeaturesAligner](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

ModelToMeasured	The transformation from the model data to the measured data.
MeasuredToModel	The transformation from the measured data to the model data.

6.4. EAngleUnit Enum

The angle units that are supported by Open eVision.

Namespace: Euresys.Open_eVision_2_10

Revolutions	Revolutions (0..1 corresponds to a full revolution).
Radians	Radians (0..2Pi corresponds to a full revolution).
Degrees	Degrees (0..360 corresponds to a full revolution).
Grades	Grades (0..400 corresponds to a full revolution).

6.5. EArithmeticLogicOperation Enum

Supported arithmetic or logic pixel-wise operators.

Namespace: Euresys.Open_eVision_2_10

Copy	Sheer copy.
Invert	Complement. (*)
Add	Saturated addition. (*)
Subtract	Saturated subtraction. (*)
Multiply	Saturated multiplication. (*)
Divide	Saturated division. (*)
Modulo	Modulo. (*)
ShiftLeft	Arithmetic left shift (multiplication by a power of 2). (*)
ShiftRight	Arithmetic right shift (division by a power of 2). (*)
ScaledAdd	Non saturating addition $((\text{left} + \text{right}) / 2)$. (*)
ScaledSubtract	Non saturating subtraction $((\text{left} + \text{complemented right}) / 2)$. (*)
ScaledMultiply	Non saturating multiplication $(\text{left} * \text{right} / 256)$ in the BW8 case, and $\text{left} * \text{right} / 65,536$ in the BW16 one). (*)
ScaledDivide	Non saturating division $(256 * \text{left} / \text{right})$ in the BW8 case, and $65,536 * \text{left} / \text{right}$ in the BW16 one). (*)
BitwiseAnd	Bitwise AND.
BitwiseOr	Bitwise OR.

BitwiseXor	Bitwise exclusive OR.
LogicalAnd	Logical AND (non zero is TRUE). (*)
LogicalOr	Logical OR (non zero is TRUE). (*)
LogicalXor	Logical exclusive OR (non zero is TRUE). (*)
Min	Minimum. (*)
Max	Maximum analysis mode.
SetZero	Copy the right operand where the left operand is zero.
SetNonZero	Copy the right operand where the left operand is non zero.
Equal	Equality comparison. (*)
NotEqual	Non equality comparison. (*)
GreaterOrEqual	"Greater or equal" comparison. (*)
LesserOrEqual	"Lesser or equal" comparison.
Greater	"Greater" comparison. (*)
Lesser	"Lesser" comparison. (*)
Compare	Absolute value of the difference. (*)
Overlay	Overlay of one image onto a source image giving a destination image. (See note at the end of the topic). (*) (**)

BitwiseNot	Same as Invert .
Average	Undefined pixels are filled using the average of their neighbours.

Remarks

(*) Not applicable for the **BW1** images/ROIs. (**) In the overlay image, black pixels (0 valued) are considered as transparent. If a **C24** image is used as overlay, all pixels (but the black ones) will be copied to the destination image. If a **BW8** image is used as overlay, all non-black pixels will be converted to the color defined by the **OverlayColor** parameter before copy to the destination image. The destination image is always a **C24** image. If no source image is given (only overlay and destination), the destination image is considered as the source image.

6.6. EasyOCR2CharacterFilter Enum

This enumeration contains the possible filters for loading fonts in [EOCR2](#).

Namespace: Euresys.Open_eVision_2_10

ASCII	All ASCII characters are loaded.
Letters	Only (alphabetic) letters are loaded.
Digits	Only digits are loaded.

6.7. EasyOCR2CharSpacingBias Enum

This enumeration contains the possible biases for the optimised character spacing in the detection phase of [EOCR2](#).

Namespace: Euresys.Open_eVision_2_10

Wide	The optimisation is biased toward wide boxes.

Neutral	The optimisation is not biased.
Narrow	The optimisation is biased toward narrow boxes.

6.8. EasyOCR2CharWidthBias Enum

This enumeration contains the possible biases for the optimised character width in the detection phase of [EOCR2](#).

Namespace: Euresys.Open_eVision_2_10

Widest	The optimisation is biased toward very wide boxes.
Wide	The optimisation is biased toward wide boxes.
Neutral	The optimisation is not biased.
Narrow	The optimisation is biased toward narrow boxes.
Narrowest	The optimisation is biased toward very narrow boxes.

6.9. EasyOCR2DrawDetectionStyle Enum

This enumeration contains the possible drawing styles for the detection results in [EOCR2](#).

Namespace: Euresys.Open_eVision_2_10

DrawChars	A bounding box is drawn around each individual detected character

DrawWords	A bounding box is drawn around each detected word, containing all characters in the word
DrawLines	A bounding box is drawn around each detected line, containing all characters/words in the line
DrawText	A bounding box is drawn around the detected text, containing all characters/words/lines in the text

6.10. EasyOCR2DrawRecognitionStyle Enum

This enumeration contains the possible drawing styles for the recognition results in [EOCR2](#).

Namespace: Euresys.Open_eVision_2_10

LeftTop	The recognition result is drawn at the left top of the character
LeftMiddle	The recognition result is drawn at the left middle of the character
LeftBottom	The recognition result is drawn at the left bottom of the character
BottomLeft	The recognition result is drawn at the bottom left of the character
BottomMiddle	The recognition result is drawn at the bottom middle of the character
BottomRight	The recognition result is drawn at the bottom right of the character
RightBottom	The recognition result is drawn at the right bottom of the character

RightMiddle	The recognition result is drawn at the right middle of the character
RightTop	The recognition result is drawn at the right top of the character
TopRight	The recognition result is drawn at the top right of the character
TopMiddle	The recognition result is drawn at the top middle of the character
TopLeft	The recognition result is drawn at the top left of the character

6.11. EasyOCR2DrawSegmentationStyle Enum

This enumeration contains the possible drawing styles for the segmentation results in [EOCR2](#).

Namespace: Euresys.Open_eVision_2_10

DrawBlobs	The segmented blobs are drawn directly.
-----------	---

6.12. EasyOCR2TextPolarity Enum

This enumeration contains the possible polarities of text searched during segmentation in [EOCR2](#).

Namespace: Euresys.Open_eVision_2_10

WhiteOnBlack	The characters appear lighter than the background.
BlackOnWhite	The characters appear darker than the background.

6.13. EAxisSystemType Enum

-

Namespace: Euresys.Open_eVision_2_10.Easy3D

CornerUpperLeft	-
CornerLowerLeft	-
Unknown	Calibration type is not defined.

6.14. ECalibrationMode Enum

Allowed values for the calibration mode.

Namespace: Euresys.Open_eVision_2_10

Raw	No calibration at all.
Inverse	The ordinate axis points downwards instead of upwards.
Scaled	The pixels are assigned a physical size.
Anisotropic	The physical size of the pixels differ horizontally and vertically. (Beware there is a restriction pertaining to the allowed image anisotropy. The resulting pixels aspect ratio should be in the range $[-4/3, -3/4]$ (or $[3/4, 4/3]$), otherwise the calibration process could fail).
Skewed	The coordinate axis make an angle with the image edge.

Tilted	The field-of-view plane is not perpendicular to the optical axis.
Radial	The lens introduces some amount of radial distortion.
Bilinear	This mode can not be combined with other calibration mode. The bilinear calibration is based on a first order polynomial approach.
Quadratic	This mode can not be combined with other calibration mode. The quadratic calibration is based on a second order polynomial approach.

6.15. ECalibrationType Enum

The Easy3D calibration type.

Namespace: Euresys.Open_eVision_2_10.Easy3D

Scale	Calibration type is EScaleCalibrationModel .
ExplicitGeometric	Calibration type is EExplicitGeometricCalibrationModel .
ObjectBased	Calibration type is EObjectBasedCalibrationModel .
Unknown	Calibration type is not defined.

6.16. ECannyThresholdingMode Enum

The thresholding modes for the Canny edge detector.

Namespace: Euresys.Open_eVision_2_10

Relative	Relative threshold; determines the required threshold level so that a given fraction of the image pixels lie below it.
Absolute	The absolute value is used.

6.17. ECC000Family Enum

-

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

ECC000	-
ECC050	-
ECC080	-
ECC100	-
ECC140	-
Unknown	Calibration type is not defined.

6.18. ECharCreationMode Enum

Allowed values for the character creation mode in EasyOCV.

Namespace: Euresys.Open_eVision_2_10

Group	Form a single character comprising all blobs.
Overlap	Form single characters out of blobs whose bounding box overlap.
Separate	Form one character for each blob.

6.19. EClippingMode Enum

Allows to choose how the fitted segment length and centre are computed

Namespace: Euresys.Open_eVision_2_10

CenteredNominal	Regular mode: the fitted segment always has the nominal length of the line gauge.
ClippedToValidSamples	The fitted segment does not extend beyond valid samples. It is clipped to the projection of the valid samples on the fitted line.
ClippedInNominalShape	The segment is built by clipping the fitted line in the rectangular range where the ELineGauge looks for valid transition samples, i.e. the rectangle which is centered and aligned on the ELineGauge nominal line, and which height is two times the ELineGauge::Tolerance .

6.20. EColorQuantization Enum

Allowed values for the quantization mode in EasyColor.

Namespace: Euresys.Open_eVision_2_10

FullRange	Values are quantized in range 0..255 .
Ccir601	Values are quantized in range 16..235 for the R, G, B or Y component, and in range 16..240 for the I, Q, U and y components.

Remarks

When quantizing the color values for the RGB or YIQ/YUV representation, one usually uses the full **0..255** range. Anyway, the CCIR has defined an alternate convention such that some values in this interval are reserved. Before performing a conversion, functions [EasyColor::SrcQuantization](#) and [EasyColor::DstQuantization](#) can be used to specify the rule used.

6.21. EColorRampMode Enum

This enumeration contains the possible values for the parameter of [E3DViewer::GenerateColors](#) method.

Namespace: Euresys.Open_eVision_2_10.Easy3D

HueFromZ	The Hue color component calculated from Z coordinate. The colors range from Red (minimum Z) to Blue (maximum Z).
RGBCube	RGB colors directly calculated from XYZ coordinates.

6.22. EColorSystem Enum

The color systems that are supported by Open eVision.

Namespace: Euresys.Open_eVision_2_10

NoColor	Undefined
Bilevel	Binary black & white.
GrayLevel	Continuous tone black & white.
Xyz	CIE XYZ.
Rgb	NTSC/PAL/SMPTE Red, Green, Blue.
Lab	CIE Lightness, a^* , b^* .
Luv	CIE Lightness, u^* , v^* .
Yuv	CCIR Luma, U Chroma, V Chroma.
Yiq	CCIR Luma, Inphase, Quadrature.
Lch	Lightness, Chroma, Hue.
Ish	Intensity, Saturation, Hue
Lsh	Lightness, Saturation, Hue.
Vsh	Value, Saturation, Hue.

Ysh	CCIR Luma, Saturation, Hue.
-----	-----------------------------

Remarks

Open eVision supports several color systems. The achromatic ones are related to black and white and gray-level images ([EImageBW1](#) and [EImageBW8](#)). The remaining ones apply to color images ([EImageC24](#)). Also see unquantized and quantized colors for the allowed ranges of values.

6.23. EConnexity Enum

Possible values for the connexity of a contour.

Namespace: Euresys.Open_eVision_2_10

Connexity4	Pixels touching by an edge are considered connected.
Connexity8	Pixels touching by an edge or a corner are considered connected.

6.24. EContourMode Enum

Possible modes for contour traversal.

Namespace: Euresys.Open_eVision_2_10

Clockwise	The contour is traversed clockwise.
ClockwiseAlwaysClosed	The contour is traversed clockwise and the image border may be followed if necessary to close the contour.
ClockwiseContinuelfBorder	Contour traversal is restarted counterclockwise when an image border is met.

Anticlockwise	The contour is traversed counterclockwise.
AnticlockwiseContinuelfBorder	Contour traversal is restarted clockwise when an image border is met.
AnticlockwiseAlwaysClosed	The contour is traversed anticlockwise and the image border may be followed if necessary to close the contour.

6.25. EContourThreshold Enum

Allowed thresholding modes for contour traversal.

Namespace: Euresys.Open_eVision_2_10

Above	Traverse the pixels just above the threshold.
Below	Traverse the pixels just below the threshold.

6.26. ECorrelationMode Enum

Allowed values for the EasyMatch correlation mode.

Namespace: Euresys.Open_eVision_2_10

Standard	Allows positioning the shape edges symmetrically.
OffsetNormalized	Correlation made insensitive to changes in intensity (computed from the centered image/pattern gray values).
GainNormalized	

Normalized

Correlation made insensitive to changes in contrast (computed from the reduced image/pattern gray values).

ray-level correlation method is used.

6.27. EDataSize Enum

Possible data sizes for an object feature.

Namespace: Euresys.Open_eVision_2_10

BitsPerPixel1	bit
BitsPerPixel8	byte
BitsPerPixel16	word
BitsPerPixel32	long word
BitsPerPixel64	quad word
BitsPerPixel24	3 bytes
BitsPerPixel96	12 bytes

6.28. EDataType Enum

Possible data types for an object feature.

Namespace: Euresys.Open_eVision_2_10

UnsignedInt	Unsigned integer.
SignedInt	Signed integer.
Float	Floating-point.

6.29. EDegreesOfFreedom Enum

Allowed values for the degrees of freedom in [EChecker](#).

Namespace: Euresys.Open_eVision_2_10

Translation	Translation allowed.
Rotation	Rotation allowed.
Scaling	Scaling allowed.

6.30. EDiagnostic Enum

Possible defect codes in EasyOCV.

Namespace: Euresys.Open_eVision_2_10

CharNotFound	Insufficient location score for a character.
CharOverprinting	Too much inking on a character.
CharUnderprinting	Too little inking on a character.

CharMismatch	Wrong character shape.
TextNotFound	Insufficient location score for a text.
TextOverprinting	Too much inking on a text.
TextUnderprinting	Too little inking on a text.
TextMismatch	Wrong text shape.
BadContrast	Global contrast (of inspection ROI) out of range.
Undefined	-

6.31. EDoubleThresholdMode Enum

The double threshold mode for the selection of coded elements with respect to a given feature.

Namespace: Euresys.Open_eVision_2_10

Inside	The outer edges of the frame remain totally inside the ROI.
Outside	The inner edges of the frame remain totally outside the ROI.

6.32. EDraggingMode Enum

Defines how the shape could be dragged

Namespace: Euresys.Open_eVision_2_10

Standard	Allows positioning the shape edges symmetrically.
ToEdges	Allows positioning each shape edge individually.

6.33. EDragHandle Enum

Allowed values for a handle identifier in the context of handle dragging.

Namespace: Euresys.Open_eVision_2_10

NoHandle	No handle.
Inside	The outer edges of the frame remain totally inside the ROI.
North	Northern handle.
East	Eastern handle.
South	Southern handle.
West	Western handle.
NorthWest	North-Western handle.
SouthWest	South-Western handle.
NorthEast	North-Eastern handle.
SouthEast	South-Eastern handle.
Center	Circle, rectangle or wedge center handle.

Org	Line segment or circle arc origin handle.
Mid	Line segment or circle arc middle handle.
End	Line segment or circle arc end handle.
Tol0	First tolerance handle.
Tol1	Second tolerance handle.
Tol_x0	Rectangle leftmost first tolerance handle.
Tol_x1	Rectangle leftmost second tolerance handle.
Tol_y0	Rectangle lower first tolerance handle.
Tol_y1	Rectangle lower second tolerance handle.
Tol_XX0	Rectangle rightmost first tolerance handle.
Tol_XX1	Rectangle rightmost second tolerance handle.
Tol_YY0	Rectangle upper first tolerance handle.
Tol_YY1	Rectangle upper second tolerance handle.
Tol_a0	Wedge leftmost first tolerance handle.
Tol_a1	Wedge leftmost second tolerance handle.
Tol_AA0	Wedge rightmost first tolerance handle.
Tol_AA1	Wedge rightmost second tolerance handle.

Tol_r0	Wedge inner first tolerance handle.
Tol_r1	Wedge inner second tolerance handle.
Tol_RR0	Wedge outer first tolerance handle.
Tol_RR1	Wedge outer second tolerance handle.
Edge_x	Rectangle leftmost edge handle.
Edge_XX	Rectangle rightmost edge handle.
Edge_y	Rectangle lower edge handle.
Edge_YY	Rectangle upper edge handle.
Edge_a	Wedge leftmost edge handle.
Edge_AA	Wedge rightmost edge handle.
Edge_r	Wedge outer edge handle.
Edge_RR	Wedge inner edge handle.

6.34. EDrawableFeature Enum

The various features that can be drawn for coded elements.

Namespace: Euresys.Open_eVision_2_10

BoundingBox	3D oriented bounding box of the object.
-------------	---

ConvexHull	The convex hull.
Ellipse	Long axis of the ellipse of inertia. (<i>Float</i>). (*)
FeretBox22	The Feret box oriented at 22.5°.
FeretBox45	The Feret box oriented at 45°.
FeretBox68	The Feret box oriented at 67.5°.
GravityCenter	Abscissa of the gravity center. (<i>Float</i>). (*)
MinimumEnclosingRectangle	The minimum enclosing rectangle.
FeretBox	The Feret box oriented at a fixed angle. Only available for selections of coded elements: The angle of interest is set through the EObjectSelection::FeretAngle property.
WeightedGravityCenter	The gravity center of the pixels of the attached image over the coded element. Only available for selections of coded elements: The attached image is set through the EObjectSelection::AttachedImage property.

6.35. EDrawingMode Enum

Allowed modes to draw the bounding box of a symbol.

Namespace: Euresys.Open_eVision_2_10

Nominal	Draws the nominal point location or model fitting gauge.
Actual	Draws the located point or the fitted model.

SampledPaths	Draws the sampled segments along the model.
SampledPath	Draws the sampled segment specified by ELineGauge::MeasureSample .
PointsInSkipRange	Draws the skipped sampled points in addition to the non-skipped points.
SampledPoints	Draws the sampled points along the model.
SampledPoint	Draws the sampled point specified by ELineGauge::MeasureSample .
InvalidSampledPoints	Draws the invalid sampled points along the model.
Learn	Draw the pattern learning ROI(s).
Match	Draw the pattern searching ROI(s).
Position	Draw the found pattern ROI(s).
Inspected	Draw the inspected ROI.
MaxInspected	Draw the largest possible inspected ROI.

6.36. EEditMode Enum

This enumeration is used to select which graphical interactions are allowed.

Namespace: Euresys.Open_eVision_2_10

None	The object cannot be edited.
------	------------------------------

Move	The object can be moved.
Rotate	The object can be rotated.
Stretch	The object can be stretched.
EdgeSplit	The object can have one of its edges split.
All	All transitions.

6.37. EEncodingConnexity Enum

The connexity mode for the encoding process.

Namespace: Euresys.Open_eVision_2_10

Four	Pixels touching by an edge are considered connected.
Eight	Pixels touching by an edge or a corner are considered connected.

6.38. EError Enum

Possible Open eVision error codes.

Namespace: Euresys.Open_eVision_2_10

Ok	Success
EndOfImageSequence	End of image sequence

UserDialogFailed	User dialog failed
ImageLimitsReached	Image limits reached
InvalidAsciiPadding	Invalid ASCII padding
InvalidOperation	Invalid operation - See Reference
InvalidBitsPerPixel	Invalid depth (bits per pixel) - Check type compatibility
InvalidDataType	Invalid data type - Check type compatibility
InvalidDataSize	Invalid data size - Check type compatibility
ParametersOutOfRange	Parameters out of range - See Reference
InvalidMode	Invalid mode - See Reference
EndSmallerThanStart	End smaller than start - Adjust indices
Parameter1OutOfRange	Parameter 1 out of range - See Reference
Parameter2OutOfRange	Parameter 2 out of range - See Reference
Parameter3OutOfRange	Parameter 3 out of range - See Reference
Parameter4OutOfRange	Parameter 4 out of range - See Reference
Parameter5OutOfRange	Parameter 5 out of range - See Reference
Parameter6OutOfRange	Parameter 6 out of range - See Reference
Parameter7OutOfRange	Parameter 7 out of range - See Reference

Parameter8OutOfRange	Parameter 8 out of range - See Reference
Parameter9OutOfRange	Parameter 9 out of range - See Reference
Parameter10OutOfRange	Parameter 10 out of range - See Reference
WindowsError	Out of GDI handles
InvalidPlanesPerPixel	Invalid planes per pixel - Check image type or file contents
BW1ImageExpected	1 bit black & white (BW1) image expected - Check image type or file contents
BW8ImageExpected	8 bits black & white (BW8) image expected - Check image type or file contents
BW16ImageExpected	16 bits black & white (BW16) image expected - Check image type or file contents
BW32ImageExpected	32 bits black & white (BW32) image expected - Check image type or file contents
TemplateCallNeedsSpecialization	Template call needs specialization
CannotCreateMutex	Cannot create Mutex
CannotLockMutex	Cannot lock Mutex
CannotUnlockMutex	Cannot unlock Mutex
CannotDeleteMutex	Cannot delete Mutex
TimeoutReached	Timeout reached

FunctionNotFound	Function not found
ProcessStopped	Process stopped
CopyNotAllowed	Copy not allowed
SingularMatrix	Internal error (code: 1050)
DivisionByZero	Division by zero - Check parameters
ReadOnlyProperty	This property is read-only and cannot be accessed
UndefinedProperty	This property is undefined and cannot be accessed
ItemNotFound	The data structure does not contain the specified item
NextItemNotFound	The specified item has no next sibling
InlineAssemblyDisabled	The inline assembly code for MMX is disabled in this build. This error should only be returned for builds compiled with GCC, or if the symbol E_MMX is left undefined.
FileAccessProblems	File access problems - Check file pathname and device state
FileCouldNotBeOpened	File could not be opened - Check file pathname, troubleshoot disk device
FileWhileReading	File error while reading - Check file integrity, troubleshoot disk device
FileWhileWriting	File error while writing - Check free disk space, troubleshoot disk device

BadFileFormat	Bad file format - Check file source or contents
FileCouldNotBeClosed	File could not be closed - Check free disk space, troubleshoot disk device
UnsupportedFileFormatVersion	Unsupported file format version - Upgrade to newer release
MissingOrUnsupportedFileExtension	Missing or unsupported file extension - Check file name/format match
FileIsReadOnly	File is read-only - Set to read-write or save under another name
UnsupportedObjectTypeInArchive	The archive does not contain the correct object type - Check your archiving routines
UnknownArchiveError	An unexpected error occurred during archive access
SerializerShouldBeInReadMode	Attempting to read with a serializer not open for read access
SerializerShouldBeInWriteMode	Attempting to write with a serializer not open for write access
FileExists	Attempting to overwrite an existing file while not allowed to do so
SerializerNotOpen	Attempting to use a serializer that is not correctly opened
UnrecognizedFileFormat	Unrecognized File Format
WrongColorFormatFileFormatCombination	Wrong Color Format File Format Combination

FileDoesNotExist	Attempting to open a file which doesn't exist
ObjectTooLargeToBeSerialized	Object is too large to be serialized
UnsupportedFileFormat	Unsupported File Format
UnsupportedTiffFormat	Unsupported TIFF format - Convert TIFF file
UnsupportedBmpFormat	Unsupported BMP format - Convert BMP file
InvalidPngCompression	Invalid PNG compression level
UnsupportedJpegFormat	Unsupported JPEG format - Convert JPEG file
BilevelImageExpected	Bi-level image expected - Use BW1
GrayLevelImageExpected	Gray-level image expected - Use BW8
ColorImageExpected	Color image expected - Use C24
BilevelFormatExpected	Bi-level format expected - Convert file to black & white
GrayLevelFormatExpected	Gray-level format expected - Convert file to gray shades
ColorFormatExpected	Color format expected - Convert file to true color
CannotReadJpegFile	Cannot read JPEG file - Troubleshoot disk device
CannotWriteJpegFile	Cannot write JPEG file - Troubleshoot disk device
WrongFileExtension	Wrong file extension
UnableToAllocateTemporaryMemory	

BufferTooSmall	Unable to allocate temporary memory - Fix memory leaks or release memory
buffer is too small. Supply a bigger buffer.	
UnableToAllocateMemory	Not enough memory for this allocation.
UnableToAccessImageMemory	Unable to access image memory - Load or size image
RoiTooLarge	ROI too large - Fit ROI to image
NotAValidImage	Not a valid image - Check image contents
ImagesNotSameSize	Images not of the same size - Adjust image size(s)
ImagesNotSameBitsPerPixel	Images not of the same depth (bits per pixel) - Choose compatible types
SourceImageTooSmall	Origin image too small - Use a larger image
PixelsMustHaveFiniteSize	Pixels must have finite size - Use non-zero parameters
ConstantIsNull	Constant is NULL - Use non-zero value
PixelNullEncountered	NULL pixel encountered - Avoid division by zero
ImagesMayNotOverlap	Images cannot overlap - Use distinct images
RoiOutOfImageLimits	ROI out of the top parent limits - Resize to fit in image
RoiAlreadyHasAParent	ROI already has a parent - Detach ROI first

RoiHasNoParentImage	There is no image ancestor for this ROI - Attach the ROI or one of its ancestor to an image
CannotApplyToAnImage	Cannot apply to an image - Apply to a ROI instead
UnsupportedImageType	Unsupported image type - Check type compatibility
InvalidImageType	Invalid image type - Check type compatibility
UnsupportedXserverDepth	Unsupported X server depth
InconsistentRoiHierarchy	The hierarchy of ROI has been corrupted (inconsistent parent/daughters relationship)
SourceImageTooBig	Original image too big - Use a smaller image
BW1RoiNotAligned	First bit index of an aligned ROI must be 0 - Use an aligned ROI
WrongRoiType	Wrong ROI or image type
CyclingParenthoodNotAllowed	Cycling Parenthood not allowed
WrongBitsPerRow	Bits per row must be a multiple of 32 (4 bytes) and must be enough to hold all the pixels of an image row.
MisalignedImagePtr	The supplied image pointer must be aligned to 4 bytes.
UnsupportedImageTypeConversion	Unsupported image type conversion
ImageFromFileDoesNotFitIntoROI	

PixelCoordinatesOutOfROI coordinate is outside the ROI/Image	The ROI is not the same size as the image file. When loading an image from a file into a ROI, the ROI must have the exact required size. On the other, when loading into an image object, it gets resized to the correct size.
ROIFromFileDoesNotFitIntoROI	The ROI is not the same size as the ROI previously saved. ROIs directly linked to an pixel container must have the same size as the container and have a (0, 0) origin.
ROIHasZeroArea	The ROI width and height must both be larger than 0 - resize the ROI.
RegionTooSmall	The region is too small.
ERegionHasNotBeenPrepared	The ERegion has not been prepared, please use function ERegion::Prepare().
ERegionImpossibleCopy	Cannot copy non-prepared geometrical region into a base ERegion instance.
PixelOutsidePerimeter	Pixel outside perimeter - Check pixel value
PixelInsidePerimeter	Pixel inside perimeter - Check pixel value
IsolatedPixel	Isolated pixel - Check pixel value
MaxPixelInContourReached	Maximum pixels in contour reached
NotAValidContour	Not a valid contour - Initialize using a contouring function
UnableToAccessVectorMemory	Unable to access vector memory - Check proper vector initialization

NotAValidVectorDescriptor	Not a valid vector descriptor
VectorTypesNotHist	Vector type is not histogram
NotEnoughGroupsInVector	Not enough groups in vector
InvalidVectorDataSize	Invalid vector data size
InvalidVectorDataType	Invalid vector data type
InvalidVectorType	Invalid vector type
ResultTooBigToFitInVector	Result too big to fit in vector
GroupOutOfRange	Group out of range - Adjust group index
InvalidVectorGroupLength	Invalid vector group length
InvalidNumberOfVectorElements	Invalid number of vector elements - Check proper vector initialization
VectorsNotSameSize	Vectors not of the same size - Adjust vector size(s)
UnableToAccessKernelMemory	Unable to access kernel memory - Check proper kernel initialization
NotAValidKernelDescriptor	Not a valid kernel descriptor
InvalidKernel	Invalid kernel
KernelInvalidSize	Invalid kernel size - Check proper kernel initialization
KernelNotAllocated	Kernel not allocated - Check proper kernel initialization

BadListPosition	Bad list position - Restart list traversal
ListIsEmpty	List is empty
TopOfList	Top of list - Do not traverse backwards
BotOfList	Bottom of list - Do not traverse forwards
ListError	List error
LicenseMissing	The license for this library is not granted - Launch License Manager
EasyImageLicenseMissing	The license for EasyImage is not granted - Launch License Manager
EasyColorLicenseMissing	The license for EasyColor is not granted - Launch License Manager
EasyObjectLicenseMissing	The license for EasyObject is not granted - Launch License Manager
EasyMatchLicenseMissing	The license for EasyMatch is not granted - Launch License Manager
EasyGaugeLicenseMissing	The license for EasyGauge is not granted - Launch License Manager
EasyFindLicenseMissing	The license for EasyFind is not granted - Launch License Manager
EasyOcrLicenseMissing	The license for EasyOCR is not granted - Launch License Manager
EasyOcvLicenseMissing	The license for EasyOCV is not granted - Launch License Manager
EasyBarCodeLicenseMissing	

EasyMatrixCodeLicenseMissing	The license for EasyBarCode is not granted - Launch License Manager
or EasyMatrixCode is not granted - Launch License	
EasyMatchAlignementModelLicenseMissing	The license for EasyMatch Alignment mode is not granted - Launch License Manager
EvisionStudioLicenseMissing	The license for eVison Studio is not granted - Launch License Manager
InvalidDongleIndex	The index do not match any available dongle
CannotWriteOEMKey	The OEM key cannot be set
WarpImagesTooSmall	Warp images too small - Increase image size
UnsupportedImageSize	Unknown error code
InvalidThresholdValue	The threshold value is not supported
UnknownFeature	Unknown feature - Check parameters
InvalidSelectionArgument	Invalid selection argument - Check parameters
SortListTooLong	Sort list too long
NotAValidOperationCode	Not a valid operation code
TooManyObjectsDetected	Too many objects detected - Increase MaxObjects
InvalidFeature	Invalid feature - Check parameters
FeatureNotCalculated	Feature not calculated - Call AnalyseObjects method

BadObjectNumber	Bad object number - Check parameters
NoObjectSelected	No object selected - Blob list is empty
LowThresholdHigherThanHighThreshold	Low threshold higher than high threshold - Adjust thresholds
InvalidThresholdMode	Invalid threshold mode - Use appropriate threshold setting method
NoImageAttached	No image attached to the selection - Use Attach()
OutOfContinuousMode	Invalid call out of continuous mode
InvalidImageTypeForSegmenter	The current segmenter can not cope with this type of image
LayersOverlapping	Two different layers are associated with the same layer index
EndOfIterator	The iterator has reached the end of the enumeration
NoThresholdComputedYet	The threshold valued has not been computed yet - First encode an image
FeatureNotDrawable	This kind of feature cannot be drawn
OnlyApplicableToObjectSelection	This kind of feature cannot be used out of EObjectSelection
MoreThanOneLayerEncoded	Please specify the layer index (several layers are encoded)
CodedElementNotSelected	The coded element is not present in the selection
NoPatternLearnt	No pattern learnt - Load from file or train pattern
PatternTooLarge	Pattern too large - Use a smaller one

PatternTooSmall	Pattern too small - Use a larger one
NotAnEasyMatchFile	Not an EasyMatch file - Check file source
UnsupportedEasyMatchFileVersion	Unsupported EasyMatch file version - Upgrade to a newer release
NoImageLearnt	No image learnt - Call LearnImage() first
WrongNumberOfDegreesOfFreedom	The number of degrees of freedom must be at least one, and no more than five - Use a value in this range
InsufficientContrast	Not enough feature points - Use a more contrasted pattern or reduce the Don't Care mask
PatternTooCloseToImageBorder	Pattern is too close to image border - Leave a margin around the pattern
IncompatibleModes	Incompatible modes (CoarseToFineAnalysisMode and PatternType)
AllowancesAndPatterntypeNotCompatible	Angle and Scale allowances can not be used with the current pattern type
ModelNotSuitedForContrastingregions	The model is unsuitable for ContrastingRegions pattern type - Try an other pattern type or increase surface of region(s)
ModelNotSuitedForConsistentedges	The model is unsuitable for ConsistentEdges pattern type - Try another pattern type or increase the model surface
NoPatternsLoaded	No patterns loaded - Load font file or train
NoPatternsInTheseClasses	No patterns in these classes - Check pattern and text class assignments
CharacterTooSmall	Character too small - Enlarge to font size

CharacterCodeTooBig8	Character code too big to fit in a string, use ReadTextWide instead
CharacterCodeTooBig16	Character code too big to fit in a wide string, use GetFirstCharCode instead
InvalidTextStructure	Text parameter doesn't fit the text topology
InvalidFontFile	The specified font file couldn't be loaded
InvalidTopology	The specified topology is invalid
InvalidEOCR2File	The file-type and structure could not be verified
EOCR2InvalidCharWidth	Character widths must be larger than 0
EOCR2InvalidCharWidthTolerance	Character width tolerance must be between 0 and 1
EOCR2InvalidCharHeight	Character height must be larger than 0
EOCR2InvalidMaxVariation	The 'maximum variation' parameter must be between 0 and 1.
EOCR2InvalidDetectionDelta	The 'detection delta' parameter must be between 0 and 128.
EOCR2InvalidMaxFragmentation	The 'maximum fragmentation' parameter must be between 0 and 1.
EOCR2InvalidSpaceWidth	Space widths must be larger than or equal to 0.
EOCR2InvalidNumDetectionPasses	NumDetectionPasses must be either 1 or 2.
EOCR2CharCodeNotSet	Character code not set
EOCR2CharHeightNotSet	Character height not set

EOCR2CharWidthNotSet	Character width not set
EOCR2TopologyNotSet	Topology not set
EOCR2DetectionFailed	The given topology and parameters could not be fitted to the detected blobs.
MismatchingColorSystem	Mismatching color system - Check transform compatibility
ColorLookupMustBeInitialized	Color lookup must be initialized - Use initialization method
UnsupportedColorTransform	Unsupported color transform
UnknownSymbolSize	Unknown symbol size - Check size initialization
UnknownEccFamily	Unknown ECC family (ECC 000/050/080/100/140/200 only)
UncorrectableErrors	Too many errors, cannot correct contents - See Reference
CouldNotLocateSymbol	Could not locate the dot matrix symbol (no good candidate object) - See Reference
UnknownFormatId	Unknown Format ID in ECC 000-140 symbol (Base 11/27/41/37 and ASCII 7/8 only) - See Reference
InvalidCrc	Invalid CRC after error correction in ECC 000-140 symbol - See Reference
NoCodeFound	Could not find any codes in the image
TimeoutReachedAndNoCodeFound	Could not find any codes in the image within the timeout
CouldNotDecodeSymbol	Could not decode symbol - Try to improve image quality

CouldNotGrade	Could not grade symbol - Quiet zone out of bounds
CouldNotLocateBarcode	Could not locate bar code symbol - Improve contrast, avoid clutter
UnrecognizedSignature	Unrecognized signature - Check enabled symbologies
InvalidNumberOfBars	Invalid number of bars - Improve bar/space contrast
ExtraEdgesFound	Extra edges found - Improve bar/space contrast or uniformity
IncoherentBarSpaceThickness	Incoherent bar/space thickness sequence - Check enabled symbologies
InvalidCheckCharacter	Invalid checksum character - Check enabled symbologies
SymbologyNotEnabled	Symbology not enabled - Invoke method SetSymbologies()
NoEdgesFound	No edges found - Adjust location or improve bar/space contrast
InvalidEMailBarcodeReaderFile	The file-type and structure could not be verified
NotAnEasyOcvFile	Not an EasyOCV file - Check file source
UnsupportedEasyOcvFileVersion	Unsupported EasyOCV file version - Upgrade Open eVision
NotEnoughSampleImages	Not enough sample images - Use AddToStatistics
NotAnEcheckerFile	Not an EChecker file - Check file source
UnsupportedEcheckerFileVersion	Unsupported EChecker file version - Upgrade Open eVision
NotEnoughSamplesLearnt	Not enough samples learnt - Use UpdateStatistics

InvalidNormalizationMode	Invalid normalization mode - Check SetNormalize call
ImageNotRegistered	Image not registered - Use method Register before Learn
InvalidLearningSequence	Invalid learning sequence - Use AVERAGE followed by ABS_DEVIATION, or RMS_DEVIATION, then READY
E_ERROR_CONTRAST_TOO_LOW	Image contrast is too low
MotherAlreadyHasThisDaughter	Mother already has this daughter - Detach daughter first
ShapeAlreadyHasDaughters	Shape already has daughters - Detach daughters first
NoValidPointFound	No valid point found in the transition computation.
NotInListAttachmentMode	Not in list attachment mode - Detach daughters first
NotInIndexedAttachmentMode	Not in indexed attachment mode - Detach daughters and call SetIndexed first
UnsupportedShapeVersion	Unsupported shape version - Upgrade Open eVision
RawCalibrationMode	Raw calibration mode - Cannot be used for this operation
BadLandmarkLayout	The layout of supplied landmarks makes the calibration impossible - Check landmarks positions
IncompatibleCalibrationModes	Incompatible calibration modes - Check calibration mode categories
NotEnoughLandmarks	Not enough landmarks to calibrate - Add landmarks or check calibration mode categories
UnexpectedShapeTypeInFile	

<p>UnsupportedModelFileVersion</p> <p>model file version - Upgrade Open eVision</p>	<p>Unexpected shape type in file - Check target shape against file model root</p>
<p>CannotAttachDetachWorldShapes</p>	<p>Cannot Attach or Detach World shape - World shapes never have a mother</p>
<p>UnexpectedWorldShapeInFile</p>	<p>Unexpected World Shape in file - Check target shape against file model root</p>
<p>UnexpectedFrameShapeInFile</p>	<p>Unexpected Frame Shape in file - Check target shape against file model root</p>
<p>UnexpectedPointShapeInFile</p>	<p>Unexpected Point Shape in file - Check target shape against file model root</p>
<p>UnexpectedLineShapeInFile</p>	<p>Unexpected Line Shape in file - Check target shape against file model root</p>
<p>UnexpectedCircleShapeInFile</p>	<p>Unexpected Circle Shape in file - Check target shape against file model root</p>
<p>UnexpectedRectangleShapeInFile</p>	<p>Unexpected Rectangle Shape in file - Check target shape against file model root</p>
<p>UnexpectedWedgeShapeInFile</p>	<p>Unexpected Wedge Shape in file - Check target shape against file model root</p>
<p>UnexpectedPointGaugeInFile</p>	<p>Unexpected Point Gauge in file - Check target shape against file model root</p>
<p>UnexpectedLineGaugeInFile</p>	<p>Unexpected Line Gauge in file - Check target shape against file model root</p>

UnexpectedCircleGaugeInFile	Unexpected Circle Gauge in file - Check target shape against file model root
UnexpectedRectangleGaugeInFile	Unexpected Rectangle Gauge in file - Check target shape against file model root
UnexpectedWedgeGaugeInFile	Unexpected Wedge Gauge in file - Check target shape against file model root
UnexpectedBarCodeInFile	Unexpected Bar Code model in file - Check target shape against file model root
AnActiveCurvedEdgesRequired	At least one curved edge must be active - Activate r and/or R edges
BrokenWedgeShapeConstraints	Constraints between the geometric and the tolerance of the wedge are broken
InvalidGrid	The detected grid is invalid
InvalidSymbolSize	The detected symbol size is invalid
InvalidFixedPattern	The fixed pattern of the detected code is invalid
CalibrationModelNotDefined	A 3D calibration model is required to perform the conversion
InvalidE3DModelFile	The file-type and structure could not be verified
EmptyPointCloud	The point cloud should not be empty
WrongOrientationVector	The supplied orientation vector is not correct
InvalidE3DCalibrationGeneratorFile	The file-type and structure could not be verified

CalibrationModelNotInitialized	A 3D calibration model is not Initialized
InvalidE3DConverterFile	The file-type and structure of the E3D converter file could not be verified
InvalidE3DCalibrationFile	The file-type and structure of the E3D calibration file could not be verified
UnknownCalibrationObjectType	The calibration object type is not set
ResultOutOfTolerances	Result out of tolerances
MalformedTriangleIndexes	The triangle indexes are not correct in E3DObject
FitFailed	The 3D fit operation failed
ParamNotSet	Trying to get a parameter value that has not been set
UndefinedPixelValue	The pixel has undefined value
FindFailed	The 3D find operation failed
AlignFailed	The 3D align operation failed
WrongCalibrationParameters	Wrong calibration parameters
WrongNormalVector	Wrong normal vector
WrongNormalTolerance	Wrong normal angle tolerance
CalibrationModelNotFound	A 3D calibration model is not found

AxesNotNormal	The axis system is not normed
AxesNotRightHanded	The axis system is not righthanded
AxesNotOrthogonal	The axis system is not orthogonal
MatrixNotRigid	A matrix is not rigid
AxisSystemNotRigid	An axis system is not rigid
CoordinatesOutOfMap	The coordinates are out of the map
InvalidAxisSystem	Axis system is invalid
InvalidPointCloud	The point cloud is not valid
PointCloudOutOfRange	The point cloud is out of range
DepthMapNotCompatibleWithCalibration	The depth map point is not compatible with the calibration model (wrong association)
InvalidE3DObjectFile	The file-type and structure of the E3D object file could not be verified
Map3DConversionModeMustBeInitialized	Conversion mode must be initialized
InvalidE3DBoxFile	The file-type and structure of the E3DBox file could not be verified
DataAugmentationFailed	Data augmentation failed for an unknown reason.
InvalidInputSpecification	Invalid input specification.
LabelDoesNotExist	The label does not exist in the dataset.

ImageIsNotAPath	The image was not given as a path.
ImageDoesNotConformToInputFormat	The images do not conform to the input format of the classifier and the automatic reformat is disabled.
CannotDisableAutoreshape	The automatic procedure to make every image conform to the input specification cannot be disabled because there already are images in the dataset that do not conform to the input specification.
NoEnoughImagesToSplitDataset	There are not enough images in the dataset to perform a split such that each part has at least one image from each label of the original dataset.
NotAvailableIn32Bits	This method is not available for the 32 bits binaries of OpenCV.
DatasetIncompatibleWithClassifier	The dataset is incompatible with this classifier. A pre-trained classifier comes with some mandatory input specifications.
ClassifierCurrentlyTraining	This operation is impossible because the classifier is currently training.
EClassifierNotTraining	Cannot wait because the classifier is not training.
CannotChangeInputSpecification	A pre-trained classifier comes with some input specifications that can't be changed.
TrainingAndValidationDatasetIncompatible	The training and validation dataset have incompatible image format.
TrainingAndValidationLabelsIncompatible	The training and validation dataset have incompatible labels.
ClassifierTrainedWithIncompatibleLabels	The classifier was previously trained with incompatible labels.

CannotChangeClassifierType	The type of classifier can't be changed once the classifier is trained.
UnknownClassifierType	Unknown classifier type.
ClassifierNotTrained	The classifier is not trained.
NoGPUFound	No GPU was found.
InvalidMetrics	The classification metrics are not valid.
MetricsIncompatibleWithResult	The classification metrics are incompatible with the given result.
InvalidClassificationResult	The classification result is invalid.
HeatmapGenerationFailure	Can not generate Heatmap.
NotEnoughMemoryForTraining	There is not enough free memory on the CPU or GPU to perform the training.
NotEnoughMemoryForPrediction	There is not enough free memory on the CPU or GPU to perform a prediction.
NotEnoughMemoryForBatchPrediction	There is not enough free memory on the CPU or GPU to perform a batch prediction.
LayerOutputDisabled	The layer has its output disabled.
NotEnoughMemoryForCache	There is not enough free memory on the CPU for storing the dataset images in the cache.
FlexnetHandleInitializationFailed	Licensing handle initialization failed
FlexnetLoadingActivationLibraryFailed	Loading of the activation library failed

FlexnetInitializationActivationLibraryFailed	Initialization of activation library failed
FlexnetActivationLibraryMismatch	Activation library mismatch
FlexnetActivationLibraryUnloaded	Activation library component has been unloaded
FlexnetLicensingServiceNotInstalled	The licensing service is not installed
FlexnetNotEnoughRights	Not enough rights to talk to service
FlexnetLicenseJobCreationFailed	License job creation failed
FLEXnetLicensePromptForFileFailed	Unable to disable license finder dialog
PixelHandling	Internal error during image processing
EmptyMorphologicalKernel	Use of a morphological kernel without any element set
MatrixOperation	Internal error during matrix processing
NonSquareMatrix	The operation is only valid for square matrices
IncompatibleMatrixSizes	The sizes of the matrix are incompatible for the operation
UnderdeterminedMatrix	Unsupported operation: The matrix has less rows than columns
OverdeterminedMatrix	Unsupported operation: The matrix has more rows than columns
PointAtInfinity	Unable to apply this operation to points at infinity
NotEnoughCalibrationPoints	Not enough points for the calibration process to succeed
LineAtInfinity	Unable to apply this operation to lines at infinity

UndeterminedGeometricEntity	Undetermined geometric entity in projective geometry
NotANumber	Not a number
MetadataAlreadyExists	Metadata already exists in the metadata store
MetadataDoesNotExist	Metadata does not exist in the metadata store
InternalError_000	Internal error 0
InternalError_001	Internal error 1
InternalError_002	Internal error 2
InternalError_003	Internal error 3
InternalError_004	Internal error 4
InternalError_005	Internal error 5
InternalError_006	Internal error 6
InternalError_007	Internal error 7
InternalError_008	Internal error 8
InternalError_009	Internal error 9
InternalError_010	Internal error 10
InternalError_011	Internal error 11
InternalError_012	Internal error 12

InternalError_013	Internal error 13
InternalError_014	Internal error 14
InternalError_015	Internal error 15
InternalError_016	Internal error 16
InternalError_017	Internal error 17
InternalError_018	Internal error 18
InternalError_019	Internal error 19
InternalError_020	Internal error 20
InternalError_021	Internal error 21
InternalError_022	Internal error 22
InternalError_023	Internal error 23
InternalError_024	Internal error 24
InternalError_025	Internal error 25
InternalError_026	Internal error 26
InternalError_027	Internal error 27
InternalError_028	Internal error 28
InternalError_029	Internal error 29

InternalError_030	Internal error 30
InternalError_031	Internal error 31
InternalError_032	Internal error 32
InternalError_033	Internal error 33
InternalError_034	Internal error 34
InternalError_035	Internal error 35
InternalError_036	Internal error 36
InternalError_037	Internal error 37
InternalError_038	Internal error 38
InternalError_039	Internal error 39
InternalError_040	Internal error 40
InternalError_041	Internal error 41
InternalError_042	Internal error 42
InternalError_043	Internal error 43
InternalError_044	Internal error 44
InternalError_045	Internal error 45
InternalError_046	Internal error 46

InternalError_047	Internal error 47
InternalError_048	Internal error 48
InternalError_049	Internal error 49
InternalError_050	Internal error 50
InternalError_051	Internal error 51
InternalError_052	Internal error 52
InternalError_053	Internal error 53
InternalError_054	Internal error 54
InternalError_055	Internal error 55
InternalError_056	Internal error 56
InternalError_057	Internal error 57
InternalError_058	Internal error 58
InternalError_059	Internal error 59
InternalError_060	Internal error 60
InternalError_061	Internal error 61
InternalError_062	Internal error 62
InternalError_063	Internal error 63

InternalError_064	Internal error 64
InternalError_065	Internal error 65
InternalError_066	Internal error 66
InternalError_067	Internal error 67
InternalError_068	Internal error 68
InternalError_069	Internal error 69
InternalError_070	Internal error 70
InternalError_071	Internal error 71
InternalError_072	Internal error 72
InternalError_073	Internal error 73
InternalError_074	Internal error 74
InternalError_075	Internal error 75
InternalError_076	Internal error 76
InternalError_077	Internal error 77
InternalError_078	Internal error 78
InternalError_079	Internal error 79
InternalError_080	Internal error 80

InternalError_081	Internal error 81
InternalError_082	Internal error 82
InternalError_083	Internal error 83
InternalError_084	Internal error 84
InternalError_085	Internal error 85
InternalError_086	Internal error 86
InternalError_087	Internal error 87
InternalError_088	Internal error 88
InternalError_089	Internal error 89
InternalError_090	Internal error 90
InternalError_091	Internal error 91
InternalError_092	Internal error 92
InternalError_093	Internal error 93
InternalError_094	Internal error 94
InternalError_095	Internal error 95
InternalError_096	Internal error 96
InternalError_097	Internal error 97

InternalError_098	Internal error 98
InternalError_099	Internal error 99
InternalError_100	Internal error 100
CannotTraceErrors	Cannot trace errors because of a system failure
NotImplemented	Feature not implemented
NullPointer	The supplied pointer is NULL
InvalidTimeout	The current timeout value is 0
InvalidTimeoutReentrancy	Cannot Stop a timeout that has not been started. Cannot Pop a timeout that has not been pushed
InvalidTimeoutState	Cannot Start a timeout that has been reached Cannot Pop a timeout that is Active
Unknown	Calibration type is not defined.

6.39. EFamily Enum

Allowed values for the ECC symbol family in EasyMatrixCode.

Namespace: Euresys.Open_eVision_2_10

ECC000	-
ECC050	-

ECC080	-
ECC100	-
ECC140	-
ECC200	ECC 200, 20 % error recovery capability.
Unknown	Calibration type is not defined.

Remarks

6.40. EFeature Enum

The various features that can be measured on the coded elements of a selection.

Namespace: Euresys.Open_eVision_2_10

ElementIndex	Index of the coded element (cf. ECodedElement::ElementIndex).
LayerIndex	Index of the layer of the coded element (cf. ECodedElement::LayerIndex).
RunCount	Number of runs (cf. ECodedElement::RunCount).
Area	Object area in metric units.
LargestRun	Size of the longest run. (<i>Signed Integer</i>).
ContourX	Starting point abscissa of the object contour. (<i>Signed Integer</i>).
ContourY	Starting point ordinate of the object contour. (<i>Signed Integer</i>).

LeftLimit	Abscissa of the leftmost pixel (cf. ECodedElement::LeftLimit).
RightLimit	Abscissa of the rightmost pixel (cf. ECodedElement::RightLimit).
TopLimit	Abscissa of the topmost pixel (cf. ECodedElement::TopLimit).
BottomLimit	Ordinate of the bottommost pixel (cf. ECodedElement::BottomLimit).
GravityCenterX	Abscissa of the gravity center. (<i>Float</i>). (*)
GravityCenterY	Ordinate of the gravity center. (<i>Float</i>). (*)
BoundingBoxCenterX	Abscissa of the center of the bounding box (cf. ECodedElement::BoundingBoxCenterX).
BoundingBoxCenterY	Ordinate of the center of the bounding box (cf. ECodedElement::BoundingBoxCenterY).
BoundingBoxWidth	Width of the bounding box (Feret diameter 0° - cf. ECodedElement::BoundingBoxWidth).
BoundingBoxHeight	Height of the bounding box (Feret diameter 90° - cf. ECodedElement::BoundingBoxHeight).
FeretBox22CenterX	Abscissa of the center of the Feret box oriented at 22.5° (cf. ECodedElement::FeretBox22CenterX).
FeretBox22CenterY	Ordinate of the center of the Feret box oriented at 22.5° (cf. ECodedElement::FeretBox22CenterY).
FeretBox22Width	Width of the Feret box oriented at 22.5° (Feret diameter at 22.5° - cf. ECodedElement::FeretBox22Width).
FeretBox22Height	

FeretBox45CenterX	Height of the Feret box oriented at 22.5° (Feret diameter at 112.5° - cf. ECodedElement::FeretBox22Height).
FeretBox45CenterY	Ordinate of the center of the Feret box oriented at 45° (cf. ECodedElement::FeretBox45CenterY).
FeretBox45Width	Width of the Feret box oriented at 45° bounding box (Feret diameter at 45° - cf. ECodedElement::FeretBox45Width).
FeretBox45Height	Height of the Feret box oriented at 45° (Feret diameter at 135° - cf. ECodedElement::FeretBox45Height).
FeretBox68CenterX	Abscissa of the center of the Feret box oriented at 67.5° (cf. ECodedElement::FeretBox68CenterX).
FeretBox68CenterY	Ordinate of the center of the Feret box oriented at 67.5° (cf. ECodedElement::FeretBox68CenterY).
FeretBox68Width	Width of the Feret box oriented at 67.5° (Feret diameter at 67.5° - cf. ECodedElement::FeretBox68Width).
FeretBox68Height	Height of the Feret box oriented at 67.5° (Feret diameter at 157.5° - cf. ECodedElement::FeretBox68Height).
MinimumEnclosingRectangleCenterX	Abscissa of the Minimum Enclosing Rectangle center (cf. ECodedElement::MinimumEnclosingRectangleCenterX).
MinimumEnclosingRectangleCenterY	Ordinate of the Minimum Enclosing Rectangle center (cf. ECodedElement::MinimumEnclosingRectangleCenterY).
MinimumEnclosingRectangleWidth	Width of the Minimum Enclosing Rectangle (cf. ECodedElement::MinimumEnclosingRectangleWidth).

MinimumEnclosingRectangleHeight	Height of the Minimum Enclosing Rectangle (cf. ECodedElement::MinimumEnclosingRectangleHeight).
MinimumEnclosingRectangleAngle	Direction of the Minimum Enclosing Rectangle (cf. ECodedElement::MinimumEnclosingRectangleAngle).
SigmaX	Centered moment of inertia around X (average squared X-deviation). (<i>Float</i>).
SigmaY	Centered moment of inertia around Y (average squared Y-deviation). (<i>Float</i>).
SigmaXX	Reduced, centered moment of inertia (around the principal inertia axis). (<i>Float</i>).
SigmaXY	Centered cross moment of inertia (average X-deviation * Y-deviation). (<i>Float</i>).
SigmaYY	Reduced, centered moment of inertia (around the secondary inertia axis). (<i>Float</i>).
EllipseWidth	Long axis of the ellipse of inertia. (<i>Float</i>). (*)
EllipseHeight	Short axis of the ellipse of inertia. (<i>Float</i>). (*)
EllipseAngle	Direction of the principal axis of inertia. (<i>Float</i>). (*)
Eccentricity	Eccentricity of the ellipse of inertia (cf. ECodedElement::Eccentricity).
FeretBoxCenterX	Abscissa of the center of the Feret box oriented at a fixed angle (cf. ECodedElement::ComputeFeretBox). The angle of interest is set through EObjectSelection::FeretAngle .
FeretBoxCenterY	

FeretBoxWidth	Ordinate of the center of the Feret box oriented at a fixed angle (cf. ECodedElement::ComputeFeretBox). The angle of interest is set through EObjectSelection::FeretAngle .
FeretBoxHeight	Height of the Feret box oriented at a fixed angle (cf. ECodedElement::ComputeFeretBox). The angle of interest is set through EObjectSelection::FeretAngle .
PixelMin	Minimum gray level of all pixels. (<i>Signed Integer</i>).
PixelMax	Maximum gray level of all pixels. (<i>Signed Integer</i>).
WeightedGravityCenterX	Abscissa of the gravity center of the pixels of the attached image over the coded element (cf. ECodedElement::ComputeWeightedGravityCenter). The attached image is set through EObjectSelection::AttachedImage .
WeightedGravityCenterY	Ordinate of the gravity center of the pixels of the attached image over the coded element (cf. ECodedElement::ComputeWeightedGravityCenter). The attached image is set through EObjectSelection::AttachedImage .
PixelGrayAverage	Average gray-level value of the object pixels. (<i>Float</i>).
PixelGrayVariance	Variance of the gray-level value of the object pixels. (<i>Float</i>).
PixelGrayDeviation	Standard deviation of the gray-level value of the attached image over the coded element (cf. ECodedElement::ComputePixelGrayDeviation). The attached image is set through EObjectSelection::AttachedImage .

6.41. EFillUndefinedPixelsDirection Enum

Direction in which the undefined pixels are filled in a depthmap.

Namespace: Euresys.Open_eVision_2_10.Easy3D

Vertical	Undefined pixels are filled using a vertical scan.
Horizontal	Undefined pixels are filled using an horizontal scan.
Combined	Undefined pixels are filled using both a vertical and an horizontal scan.
Local	Specialized method for filling undefined pixels. Undefined pixels are filled using their 4 neighboring pixels: If at least 2 out of 4 neighbors have a 'defined' value, the pixel will be filled with their average value. Else, the pixel will remain 'undefined'.

6.42. EFillUndefinedPixelsMethod Enum

Method to fill the undefined pixels in a depthmap.

Namespace: Euresys.Open_eVision_2_10.Easy3D

KeepMinimum	Undefined pixels are filled using the minimum of their neighbours.
KeepMaximum	Undefined pixels are filled using the maximum of their neighbours.
Average	Undefined pixels are filled using the average of their neighbours.

Ramp

Undefined pixels are filled using a ramp between their neighbours.

6-

.-

43. EFilteringMode Enum

Allowed values for the filtering mode of EasyMatch.

Namespace: Euresys.Open_eVision_2_10

Uniform	-
LowPass	Filtering with a low-pass 3x3 kernel. This is the preferred mode for images featuring sharp gray-level transitions.

6.44. EFindContrastMode Enum

Allowed values for the contrast mode of EasyMatch.

Namespace: Euresys.Open_eVision_2_10

Normal	Normal contrast. Pattern occurrences will be found with the same contrast as at learn time. Default mode.
Inverse	The ordinate axis points downwards instead of upwards.

Any	The operation applies to both selected and unselected items.
PointByPointNormal	-
PointByPointInverse	-
PointByPointAny	-
Unknown	Calibration type is not defined.

6.45. EFlipping Enum

Allowed values for the symbol flipping type in EasyMatrixCode.

Namespace: Euresys.Open_eVision_2_10

Yes	Image is flipped.
No	Image is not flipped.
Unknown	Calibration type is not defined.

6.46. EFramePosition Enum

This enumeration contains the possible values for the placement of the overlay frame edges that are drawn to highlight the position of an ROI.

Namespace: Euresys.Open_eVision_2_10

--

On	The frame is centered on the ROI edges.
Inside	The outer edges of the frame remain totally inside the ROI.
Outside	The inner edges of the frame remain totally outside the ROI.

6.47. EGrayscaleSingleThreshold Enum

The modes that are available to segment a grayscale image using a single threshold.

Namespace: Euresys.Open_eVision_2_10

Absolute	The absolute value is used.
Relative	Relative threshold; determines the required threshold level so that a given fraction of the image pixels lie below it.
MinResidue	Selects a threshold value such that the quadratic difference between the source and thresholded image is minimized.
MaxEntropy	Selects a threshold value such that the entropy (i.e. the amount of information) of the resulting thresholded image is maximized.
IsoData	Thresholds the image using an automatically-computed value halfway between the average dark gray value (i.e. gray levels below the threshold) and average light gray values (i.e. gray levels above the threshold).

6.48. EHarrisThresholdingMode Enum

The thresholding modes for the Harris corner detector.

Namespace: Euresys.Open_eVision_2_10

Relative	Relative threshold; determines the required threshold level so that a given fraction of the image pixels lie below it.
Absolute	The absolute value is used.

6.49. EHistogramFeature Enum

The various parameters that can be extracted from a histogram.

Namespace: Euresys.Open_eVision_2_10

MostFrequentPixelValue	Value of the most frequent pixel.
MostFrequentPixelFrequency	Frequency of the most frequent pixel.
LeastFrequentPixelValue	Value of the least frequent pixel.
LeastFrequentPixelFrequency	Frequency of the least frequent pixel.
SmallestPixelValue	Smallest pixel value.
GreatestPixelValue	Largest pixel value.

PixelCount	Number of pixels.
AveragePixelValue	Mean of the pixel values.
PixelValueStdDev	Standard deviation of the pixel values.

6.50. EHitAndMissValue Enum

The allowed values for the elements of a hit-and-miss kernel.

Namespace: Euresys.Open_eVision_2_10

Background	The element belongs to the background.
DontCare	The element does not belongs to the background, neither to the foreground. It is ignored by the kernel.
Foreground	The element belongs to the foreground.

6.51. EImageFileType Enum

-

Namespace: Euresys.Open_eVision_2_10

Bmp	-
Jpeg2000	-

Jpeg	-
Png	-
Tiff	-
Auto	Chooses automatically the orientation of the ZMap.
Euresys	-
Unknown	Calibration type is not defined.

6.52. EImageType Enum

Image type.

Namespace: Euresys.Open_eVision_2_10

BW1	Bi-level image.
BW8	8 bits per pixel gray-level image.
BW16	16 bits per pixel gray-level image
BW32	32 bits per pixel gray-level image.
C15	15 bits per pixel color image (R:5, G:5, B:5).
C16	16 bits per pixel color image (R:5, G:6, B:5).
C24	24 bits per pixel color image (RGB).

C24A	32 bits per pixel color image (RGB + unused alpha channel).
C48	48 bits per pixel color image (RGB).
Depth8	8 bits per pixel gray-level image.
Depth16	16 bits per pixel gray-level image.
Depth32f	32 bits per pixel gray-level image.

Remarks

For example, an [EImageC24](#) has type value [C24](#) and its pixels are typed as [EC24](#).

6.53. EKernelRectifier Enum

Possible values for the rectification mode of a kernel. This property allows specifying how negative convolution result values are handled.

Namespace: Euresys.Open_eVision_2_10

DoNotRectify	The offset of the kernel is added to the values resulting from the convolution. Negative values are then set to zero, and the values that exceed the maximum value for the image type are set to this maximum value.
KeepNegative	The positive values are discarded (set to zero) and the magnitude (absolute value) of the negative values is used.
KeepPositive	Positive value is used. The negative values are discarded (set to zero).
Absolute	The absolute value is used.

6.54. EKernelRotation Enum

Possible values for rotating a convolution kernel.

Namespace: Euresys.Open_eVision_2_10

NoRotation	No rotation of the structuring element.
Clockwise	The contour is traversed clockwise.
Anticlockwise	The contour is traversed counterclockwise.

6.55. EKernelType Enum

The types of convolution kernels that are supported by Open eVision.

Namespace: Euresys.Open_eVision_2_10

WhiteSkelet	White skeleton morphological probe.
BlackSkelet	Black skeleton morphological probe.
Edge	Edge detection morphological probe.
SobelX	X-axis Sobel derivative.
SobelY	Y-axis Sobel derivative.
PrewittX	X-axis Prewitt derivative.

PrewittY	Y-axis Prewitt derivative.
Laplacian4	4-connected Laplacian.
Laplacian8	8-connected Laplacian.
LowPass1	Low pass filter.
LowPass2	Low pass filter (average of neighbors).
LowPass3	Low pass filter (average).
HighPass1	High pass filter (value plus 4-connected Laplacian).
HighPass2	High pass filter (value plus 8-connected Laplacian).
Sobel	-
Prewitt	-
Roberts	-
Uniform3x3	-
Gaussian3x3	-
Uniform5x5	-
Gaussian5x5	-
Gaussian7x7	-
Uniform7x7	-

LaplacianX	-
LaplacianY	-
Gradient	-
GradientX	-
GradientY	-
Uniform	-
Gaussian	-

6.56. ELearningMode Enum

Allowed values for the learning mode in EasyOCV.

Namespace: Euresys.Open_eVision_2_10

Reset	Restart learning.
Template	Train the mother image.
Average	Undefined pixels are filled using the average of their neighbours.
RmsDeviation	Accumulate an image for standard deviation estimation.
AbsDeviation	Accumulate an image for robust deviation estimation.
Ready	End learning and compute threshold images.

6.57. ELearnParam Enum

Allowed values for the kind of parameters that can be learnt by EasyMatrixCode.

Namespace: Euresys.Open_eVision_2_10

LogicalSize	The data matrix code symbol logical sizes the candidate is matched against at read time.
ContrastType	The data matrix code contrast types the candidate is matched against at read time.
Flipping	The data matrix code flipping values the candidate is matched against at read time.
Family	The data matrix code families the candidate is matched against at read time.
NumItems	-

6.58. ELegacyFeature Enum

The various parameters that can be extracted from a histogram. This enumeration pertains to the EasyObject legacy API. Please use [ECodedImage2](#) instead.

Namespace: Euresys.Open_eVision_2_10

NoFeature	-

Class	Class number.
RunsNumber	Number of runs.
Area	Object area in metric units.
LargestRun	Size of the longest run. (<i>Signed Integer</i>).
GravityCenterX	Abscissa of the gravity center. (<i>Float</i>). (*)
GravityCenterY	Ordinate of the gravity center. (<i>Float</i>). (*)
LimitCenterX	Abscissa of the center of the bounding box. (<i>Float</i>). (*)
LimitCenterY	Ordinate of the center of the bounding box. (<i>Float</i>). (*)
LimitWidth	Width of the bounding box (Feret's diameter 0°). (<i>Float</i>). (*)
LimitHeight	Height of the bounding box (Feret's diameter 90°). (<i>Float</i>). (*)
Limit45CenterX	Abscissa of the center of the 45° bounding box. (<i>Float</i>). (*)
Limit45CenterY	Ordinate of the center of the 45° bounding box. (<i>Float</i>). (*)
Limit45Width	Width of the 45° bounding box (Feret's diameter 45°). (<i>Float</i>). (*)
Limit45Height	Height of the 45° bounding box (Feret's diameter 135°). (<i>Float</i>). (*)
ContourX	Starting point abscissa of the object contour. (<i>Signed Integer</i>).
ContourY	Starting point ordinate of the object contour. (<i>Signed Integer</i>).
PixelMin	Minimum gray level of all pixels. (<i>Signed Integer</i>).

PixelMax	Maximum gray level of all pixels. (<i>Signed Integer</i>).
SigmaX	Centered moment of inertia around X (average squared X-deviation). (<i>Float</i>).
SigmaY	Centered moment of inertia around Y (average squared Y-deviation). (<i>Float</i>).
SigmaXY	Centered cross moment of inertia (average X-deviation * Y-deviation). (<i>Float</i>).
SigmaXX	Reduced, centered moment of inertia (around the principal inertia axis). (<i>Float</i>).
SigmaYY	Reduced, centered moment of inertia (around the secondary inertia axis). (<i>Float</i>).
EllipseWidth	Long axis of the ellipse of inertia. (<i>Float</i>). (*)
EllipseHeight	Short axis of the ellipse of inertia. (<i>Float</i>). (*)
EllipseAngle	Direction of the principal axis of inertia. (<i>Float</i>). (*)
CentroidX	Abscissa of the weighted gravity center. (<i>Float</i>). (*)
CentroidY	Ordinate of the weighted gravity center. (<i>Float</i>). (*)
PixelGrayAverage	Average gray-level value of the object pixels. (<i>Float</i>).
PixelGrayVariance	Variance of the gray-level value of the object pixels. (<i>Float</i>).
Limit22CenterX	Abscissa of the center of the 22.5° bounding box. (<i>Float</i>). (*)
Limit22CenterY	Ordinate of the center of the 22.5° bounding box. (<i>Float</i>). (*)

Limit22Width	Width of the 22.5° bounding box (Feret's diameter 22.5°). (Float). (*)
Limit22Height	Height the 22.5° bounding box (Feret's diameter 112.5°). (Float). (*)
Limit68CenterX	Abscissa of the center of the 67.5° bounding box. (Float). (*)
Limit68CenterY	Ordinate of the center of the 67.5° bounding box. (Float). (*)
Limit68Width	Width of the 67.5° bounding box (Feret's diameter 67.5°). (Float). (*)
Limit68Height	Height of the 67.5° bounding box (Feret's diameter 157.5°). (Float). (*)
LimitAngledCenterX	Abscissa of the center of the bounding box having a skew angle defined by the LimitAngle property. (Float).
LimitAngledCenterY	Ordinate of the center of the bounding box having a skew angle defined by the LimitAngle property. (Float).
LimitAngledWidth	Width of the bounding box having a skew angle defined by the LimitAngle property (Feret's diameter [LimitAngle]). (Float).
LimitAngledHeight	Height of the bounding box having a skew angle defined by the LimitAngle property (Feret's diameter [LimitAngle + 90°]). (Float).
FeretCenterX	Abscissa of the Feret's bounding box center. (Float). (*)
FeretCenterY	Ordinate of the Feret's bounding box center. (Float). (*)
FeretWidth	Width of the Feret's bounding box. (Float). (*)
FeretHeight	Height of the Feret's bounding box. (Float). (*)

FeretAngle	Direction of the Feret's bounding box. (<i>Float</i>). (*)
ObjectNumber	Identification number.
GravityCenter	Abscissa of the gravity center. (<i>Float</i>). (*)
Limit	Abscissa of the center of the bounding box. (<i>Float</i>). (*)
Limit22	Abscissa of the center of the 22.5° bounding box. (<i>Float</i>). (*)
Limit45	Abscissa of the center of the 45° bounding box. (<i>Float</i>). (*)
Limit68	Abscissa of the center of the 67.5° bounding box. (<i>Float</i>). (*)
LimitAngled	Abscissa of the center of the bounding box having a skew angle defined by the LimitAngle property. (<i>Float</i>).
Ellipse	Long axis of the ellipse of inertia. (<i>Float</i>). (*)
Centroid	-
Feret	-

6.59. ELocalSearchMode Enum

Allowed values for the local search mode of EasyFind.

Namespace: Euresys.Open_eVision_2_10

Basic

ExtendedTranslation	Default local search neighborhood. Sets EPatternFinder::AngleSearchExtent , EPatternFinder::ScaleSearchExtent , EPatternFinder::XSearchExtent and EPatternFinder::YSearchExtent to 3.
ExtendedAll	Local search neighborhood extended on all degrees of freedom. Sets EPatternFinder::AngleSearchExtent , EPatternFinder::ScaleSearchExtent , EPatternFinder::XSearchExtent and EPatternFinder::YSearchExtent to 5.
ExtendedMore	Local search neighborhood even more extended on all degrees of freedom. Sets EPatternFinder::AngleSearchExtent and EPatternFinder::ScaleSearchExtent to 7. Sets EPatternFinder::XSearchExtent and EPatternFinder::YSearchExtent to 9.
Reserved	Reserved for internal use.
Custom	Custom local search neighborhood. Set EPatternFinder::AngleSearchExtent , EPatternFinder::ScaleSearchExtent , EPatternFinder::XSearchExtent and EPatternFinder::YSearchExtent to custom values.

6.60. ELocationMode Enum

Allowed values for the kind of pre-processing to be applied to the sample image in EasyOCV.

Namespace: Euresys.Open_eVision_2_10

Raw	No calibration at all.
Binarized	The gray-level image is thresholded before location, thus enhancing contrast between the characters and the foreground.
Gradient	-
Laplacian	The Laplacian of the image is operated on.

Remarks

Experience reveals that the best location reliability is achieved by the [Binarized](#) and [Gradient](#) modes. Additionally, the [Gradient](#) mode is not sensitive to the choice of a threshold level. Use of the [Laplacian](#) mode is not recommended.

6.61. ELogicalSize Enum

Allowed values for the logical size of Data Matrix codes in EasyMatrixCode.

Namespace: Euresys.Open_eVision_2_10

_9x9	ECC 000-140 squares.
_11x11	ECC 000-140 squares.
_13x13	ECC 000-140 squares.
_15x15	ECC 000-140 squares.
_17x17	ECC 000-140 squares.
_19x19	ECC 000-140 squares.
_21x21	ECC 000-140 squares.

_23x23	ECC 000-140 squares.
_25x25	ECC 000-140 squares.
_27x27	ECC 000-140 squares.
_29x29	ECC 000-140 squares.
_31x31	ECC 000-140 squares.
_33x33	ECC 000-140 squares.
_35x35	ECC 000-140 squares.
_37x37	ECC 000-140 squares.
_39x39	ECC 000-140 squares.
_41x41	ECC 000-140 squares.
_43x43	ECC 000-140 squares.
_45x45	ECC 000-140 squares.
_47x47	ECC 000-140 squares.
_49x49	ECC 000-140 squares.
_10x10	ECC 200 squares.
_12x12	ECC 200 squares.
_14x14	ECC 200 squares.

_16x16	ECC 200 squares.
_18x18	ECC 200 squares.
_20x20	ECC 200 squares.
_22x22	ECC 200 squares.
_24x24	ECC 200 squares.
_26x26	ECC 200 squares.
_32x32	ECC 200 squares.
_36x36	ECC 200 squares.
_40x40	ECC 200 squares.
_44x44	ECC 200 squares.
_48x48	ECC 200 squares.
_52x52	ECC 200 squares.
_64x64	ECC 200 squares.
_72x72	ECC 200 squares.
_80x80	ECC 200 squares.
_88x88	ECC 200 squares.
_96x96	ECC 200 squares.

_104x104	ECC 200 squares.
_120x120	ECC 200 squares.
_132x132	ECC 200 squares.
_144x144	ECC 200 squares.
_8x18	ECC 200 rectangles
_8x32	ECC 200 rectangles
_12x26	ECC 200 rectangles
_12x36	ECC 200 rectangles
_16x36	ECC 200 rectangles
_16x48	ECC 200 rectangles
Unknown	Calibration type is not defined.

Remarks

6.62. EMailBarcodeOrientation Enum

The orientations supported by EMailBarcode

Namespace: Euresys.Open_eVision_2_10

Unknown	Calibration type is not defined.

NoRotation	No rotation of the structuring element.
Rotated180	Upside-down barcode. Horizontal and read from right to left.
Rotated90Right	Barcode rotated 90° to the right. Vertical and read from top to bottom.
Rotated90Left	Barcode rotated 90° to the left. Vertical and read from bottom to top.
Horizontal	Undefined pixels are filled using an horizontal scan.
Vertical	Undefined pixels are filled using a vertical scan.
Any	The operation applies to both selected and unselected items.

6.63. EMailBarcodeSymbologies Enum

The symbologies supported by EMailBarcode

Namespace: Euresys.Open_eVision_2_10

Unknown	Calibration type is not defined.
JapanPost	Japan Post symbology.
Postnet	US POSTNET symbology.
Planet	US PLANET symbology.
IntelligentMail	US Intelligent Mail symbology.

USMail

US symbologies.

6-

.-

64. EMapConversionMode Enum

Conversion modes to use in [EConverter](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

MaxDynamic	Scale the value to fill the destination dynamic.
Shift	Convert value by bit shifting.

6.65. EMapConversionMode Enum

Conversion modes to use in [EConverter](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

MaxDynamic	Scale the value to fill the destination dynamic.
Shift	Convert value by bit shifting.

6.66. EMatchContrastMode Enum

Allowed values for the contrast mode of EasyMatch.

Namespace: Euresys.Open_eVision_2_10

Normal	Normal contrast. Pattern occurrences will be found with the same contrast as at learn time. Default mode.
Inverse	The ordinate axis points downwards instead of upwards.
Any	The operation applies to both selected and unselected items.

6.67. EMatchingMode Enum

Allowed values for the matching mode of EasyOCR.

Namespace: Euresys.Open_eVision_2_10

Rms	Root-mean-square error method is used.
Standard	Allows positioning the shape edges symmetrically.
Normalized	Normalized gray-level correlation method is used.

6.68. EMatrixCodeContrastMode Enum

-

Namespace: Euresys.Open_eVision_2_10

BlackOnWhite	The characters appear darker than the background.
--------------	---

WhiteOnBlack	The characters appear lighter than the background.
--------------	--

6.69. EMaximumAnalysisMode Enum

This enumeration contains the possible values for the analysis mode of the [ELaserLineExtractor](#) object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

Peaks	Peak analysis mode.
Max	Maximum analysis mode.
COG	Center of gravity analysis mode.

6.70. ENoiseRemovalMethod Enum

Type of filter used in method [EFilters::RemoveNoise](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

AbsoluteDifferenceFromMean	Removes points for which the deviation from the average height in the filter window is larger than the specified threshold. The threshold is the absolute value of the maximum difference.
RelativeDifferenceFromMean	Removes points for which the deviation from the average height in the filter window is larger than the specified threshold multiplied by the standard deviation (in the same filter window). The threshold is a factor.
HighStandardDeviation	

6.71.

ENormalizationMode Enum

Removes points for which the standard deviation calculated in the filter window is larger than a specified threshold. The threshold is the maximum standard deviation.

Allowed values for the gray-level normalization mode in EasyOCV.

Namespace: Euresys.Open_eVision_2_10

NoNormalization	No gray-level normalization (contrast changes will be detected).
Moments	Contrast normalization based on linear change.
Threshold	Contrast normalization based on non-linear change.

6.72.

EObjectBasedCalibrationPrecisionVsSpeedTradeOf f Enum

The precision vs speed tradeoff.

Namespace: Euresys.Open_eVision_2_10.Easy3D

Fast	Fast mode.
Balanced	Balanced mode.

Precise

Precise mode.

6-

.-

73. EObjectBasedCalibrationType Enum

The type of the calibration object.

Namespace: Euresys.Open_eVision_2_10.Easy3D

DoublePyramid	Double Pyramid calibration model.
TruncatedDoublePyramid	Double Truncated Pyramid calibration model.
NotDefined	Not Defined.

6.74. EOCR2DetectionMethod Enum

This enumeration contains the possible methods for text detection in [EOCR2](#).

Namespace: Euresys.Open_eVision_2_10

FixedWidth	This method is suited for detecting texts with fixed-width fonts and dotted characters.
Proportional	This method is suited for detecting texts with proportional fonts.

6.75. EOOCR2SegmentationMethod Enum

This enumeration contains the possible methods for image segmentation in [EOOCR2](#).

Namespace: Euresys.Open_eVision_2_10

Local	Specialized method for filling undefined pixels. Undefined pixels are filled using their 4 neighboring pixels: If at least 2 out of 4 neighbors have a 'defined' value, the pixel will be filled with their average value. Else, the pixel will remain 'undefined'.
Global	This method is fast and suited for segmenting images with clear text on a uniform background, it uses a global threshold value.

6.76. EOOCRClass Enum

Allowed values for the class of pattern in EasyOCR. These classes have no pre-defined meaning, the user is free to give them any meaning they like. For instance, class 0 could contains only digits, class 1 only the forward slash character, class 3 uppercase letters, etc. Any choice is allowed, as long as the correct classes are specified during the learning process. During recognition/reading, the user can specify the expected class for each character. This means that if a forward slash is expected at that position, they can say it should be class 1 (following the example from above). This will improve the recognition rate and speed because the algorithm only has to choose between characters within the specified class.

Namespace: Euresys.Open_eVision_2_10

_0	Character belongs to class 0.
_1	Character belongs to class 1.
_2	Character belongs to class 2.

_3	Character belongs to class 3.
_4	Character belongs to class 4.
_5	Character belongs to class 5.
_6	Character belongs to class 6.
_7	Character belongs to class 7.
_8	Character belongs to class 8.
_9	Character belongs to class 9.
_10	Character belongs to class 10.
_11	Character belongs to class 11.
_12	Character belongs to class 12.
_13	Character belongs to class 13.
_14	Character belongs to class 14.
_15	Character belongs to class 15.
_16	Character belongs to class 16.
_17	Character belongs to class 17.
_18	Character belongs to class 18.
_19	Character belongs to class 19.

_20	Character belongs to class 20.
_21	Character belongs to class 21.
_22	Character belongs to class 22.
_23	Character belongs to class 23.
_24	Character belongs to class 24.
_25	Character belongs to class 25.
_26	Character belongs to class 26.
_27	Character belongs to class 27.
_28	Character belongs to class 28.
_29	Character belongs to class 29.
_30	Character belongs to class 30.
Digit	Character belongs to class 0. Equivalent to _0 .
UpperCase	Character belongs to class 1. Equivalent to _1 .
LowerCase	Character belongs to class 2. Equivalent to _2 .
Special	Character belongs to class 3. Equivalent to _3 .
Extended	Character belongs to class 4. Equivalent to _4 .
AllClasses	Character belongs to all classes, from 0 to 31 included.

6.77. EOOCRColor Enum

Allowed values for the text color in EasyOCR.

Namespace: Euresys.Open_eVision_2_10

BlackOnWhite	The characters appear darker than the background.
WhiteOnBlack	The characters appear lighter than the background.
DarkOnLight	The characters appear darker than the background. No thresholding takes place when the characters are learnt and/or recognized.
LightOnDark	The characters appear lighter than the background. No thresholding takes place when the characters are learnt and/or recognized.

6.78. EPatternType Enum

Allowed values for the type of patterns in EasyFind.

Namespace: Euresys.Open_eVision_2_10

ConsistentEdges	Defines the ConsistentEdges pattern type.
ContrastingRegions	Defines the ContrastingRegions pattern type.
ThinStructure	Defines the ThinStructure pattern type.
Unknown	Calibration type is not defined.

6.79. EPickingMode Enum

-

Namespace: Euresys.Open_eVision_2_10

All	All transitions.
Begin	-
End	Line segment or circle arc end handle.
Central	-
Score	-

6.80. EPlaneCropperType Enum

Type of crop to use in [EPlaneCropper](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

KeepAbove	Only the points above the plane are selected.
KeepBelow	Only the points below the plane are selected.
KeepClose	Only the points closer to the plane than a specified distance ("maxDistance") are selected.
KeepFar	Only the points further away from the plane than a specified distance ("maxDistance") are selected.

6.81. EPlotItem Enum

Defines how the profile is drawn across a gauge.

Namespace: Euresys.Open_eVision_2_10

Transitions	Displays the profile along a point location gauge.
Peak	Displays the corresponding derivative curve.
Thresholds	Displays the threshold and minimum amplitude levels.
Points	Displays the valid transitions.

6.82. EQRCodeCodingMode Enum

This enumeration contains the possible values for the coding mode of a QR code.

Namespace: Euresys.Open_eVision_2_10

Basic	Default local search neighborhood. Sets EPatternFinder::AngleSearchExtent , EPatternFinder::ScaleSearchExtent , EPatternFinder::XSearchExtent and EPatternFinder::YSearchExtent to 3.
Fnc1_Gs1	The QR code uses the FNC1/GS1 coding mode (FNC1 in first position).
Fnc1_Aim	The QR code uses the FNC1/AIM coding mode (FNC1 in second position).

6.83. EQRCodeEncoding Enum

This enumeration contains the possible values for the encoding used for parts of the bit stream of a QR code.

Namespace: Euresys.Open_eVision_2_10

Numeric	The stream part is coded numerically.
Alphanumeric	The stream part is coded alphanumerically.
Byte	The stream part is coded as bytes.
Kanji	The stream part is coded in Kanji.

6.84. EQRCodeLevel Enum

This enumeration contains the possible values for the level of error correction of a QR code.

Namespace: Euresys.Open_eVision_2_10

L	The QR code is level L (about 7% of error correction).
M	The QR code is level M (about 15% of error correction).
Q	The QR code is level Q (about 25% of error correction).
H	The QR code is level H (about 30% of error correction).

6.85. EQRCODEMODEL Enum

This enumeration contains the possible values for a QR code model.

Namespace: Euresys.Open_eVision_2_10

Model1	The QR code is a model 1.
Model2	The QR code is a model 2 or 2005.
MicroQR	The QR code is a Micro QR.

6.86. EQRCODEPERSPECTIVEMODE Enum

This enumeration contains the possible values for the scanning precision of a QR Code reader object.

Namespace: Euresys.Open_eVision_2_10

Basic	Default local search neighborhood. Sets EPatternFinder::AngleSearchExtent , EPatternFinder::ScaleSearchExtent , EPatternFinder::XSearchExtent and EPatternFinder::YSearchExtent to 3.
Improved	The QR Code reader handles heavy perspective deformations.

Remarks

This setting has deprecated as per Open eVision release 2.0, setting the perspectiveMode will have no effect. The EQRCODEPERSPECTIVEMODE_BASIC option has been superseded by EQRDetectionMethod_GradientLegacy, the EQRCODEPERSPECTIVEMODE_IMPROVED option has been superseded by EQRDetectionMethod_PerspectiveLegacy.

6.87. EQRCodeScanPrecision Enum

This enumeration contains the possible values for the scanning precision of a QR code reader object.

Namespace: Euresys.Open_eVision_2_10

Automatic	The QR code reader determines the scan precision automatically.
Fine	The QR code reader scans finely the search field to find the QR codes. This value is recommended for small images or large images with small QR codes.
Coarse	The QR code reader scans coarsely the search field to find the QR codes. This value is recommended for large images with medium to large QR codes.

6.88. EQRDetectionMethod Enum

This enumeration contains the possible detection methods for QR codes, combinations of the methods are allowed.

Namespace: Euresys.Open_eVision_2_10

AdaptiveThreshold	This method detects finder patterns based on adaptive thresholding of the image.
Gradient	-
PerspectiveLegacy	This selects the gradient-based detection algorithms with improved perspective mode developed for eVision 1.2.2.
GradientLegacy	This selects the gradient-based detection algorithms with basic perspective mode developed for eVision 1.2.2.

Remarks

The variables Topology, CharHeight, CharWidth should be set before performing this operation.

6.89. EQRDetectionTradeOff Enum

This enumeration contains several settings for the tradeoff between detection speed and reliability of the easyQRCode methods. Setting this parameter will overwrite the current settings for EQRDetectionMethod and EQRCodeScanPrecision.

Namespace: Euresys.Open_eVision_2_10

FavorSpeed	This setting gives the fastest detection speed, but may reduce the detection accuracy. Sets EQRDetectionMethod_AdaptiveThreshold and EQRCodeScanPrecision_Coarse.
Balanced	Balanced mode.
FavorReliability	This setting gives the best detection reliability, at the cost of detection speed. Sets EQRDetectionMethod_AdaptiveThreshold EQRDetectionMethod_Gradient and EQRCodeScanPrecision_Fine.
Custom	Custom local search neighborhood. Set EPatternFinder::AngleSearchExtent , EPatternFinder::ScaleSearchExtent , EPatternFinder::XSearchExtent and EPatternFinder::YSearchExtent to custom values.

6.90. EQualityIndicator Enum

Allowed values for the quality indicators in EasyOCV.

Namespace: Euresys.Open_eVision_2_10

Location	Global scores based on the edge pixels of the characters.
Area	Object area in metric units.
Sum	Sum of the normalized foreground and background gray-level values.
Correlation	Normalized correlation.
Contrast	Global contrast of the inspected ROI.

6.91. EReadMode Enum

This enumeration contains the possible operation modes for the [EMatrixCodeReader::Read](#) method

Namespace: Euresys.Open_eVision_2_10.EasyMatrixCode2

Speed	<p>The EMatrixCodeReader::Read method will halt as soon as one of the following is true:</p> <ol style="list-style-type: none">(1) it has found the required number of codes given by EMatrixCodeReader::MaxNumCodes.(2) it reaches the timelimit given by EMatrixCodeReader::TimeOut.(3) it has completely finished its process. <p>This mode will result in the shortest processing times.</p>
Quality	<p>The EMatrixCodeReader::Read method will keep trying to improve its detection until one of the following is true:</p> <ol style="list-style-type: none">(1) it reaches the timelimit given by EMatrixCodeReader::TimeOut.(2) it has completely finished its process. <p>This mode will results in the best grading results.</p>

6.92. ERectangleMode Enum

The modes that specify how the selection of coded elements with a rectangle behaves.

Namespace: Euresys.Open_eVision_2_10

EntirelyInside	Takes into consideration only the coded elements that entirely lie inside the given rectangle, not touching its borders
EntirelyOutside	Takes into consideration only the coded elements that entirely lie outside the given rectangle, not touching its borders.
InsideOrOnBorder	Takes into consideration only the coded elements that entirely lie inside the given rectangle, or that touch its borders.
OutsideOrOnBorder	Takes into consideration only the coded elements that entirely lie outside the given rectangle, or that touch its borders.
OnBorder	Takes into consideration only the coded elements that touch the borders of the rectangle.

6.93. EReductionMode Enum

The reduction mode to be used when learning a Consistent Edges model.

Namespace: Euresys.Open_eVision_2_10

Auto	Chooses automatically the orientation of the ZMap.
Manual	Use a user-set value for the reduction strength when learning a model (cf. the property EPatternFinder::ReductionStrength).
Unknown	Calibration type is not defined.

6.94. EReferenceNoise Enum

Enumeration for specifying how a reference image is affected by noise in EasyImage.

Namespace: Euresys.Open_eVision_2_10

NoReference	The reference image is free from noise (synthetic image or noise source cancelled).
SameAsImage	The reference image is contaminated by the same noise source as the source image.

6.95. ERgbStandard Enum

Allowed values for the RGB standard in EasyColor.

Namespace: Euresys.Open_eVision_2_10

Ntsc	NTSC primaries with the following CIE XYZ coordinates: Red: (0.607, 0.299, 0.000), Green: (0.174, 0.587, 0.066), Blue: (0.201, 0.114, 1.117).
Pal	PAL primaries with the following CIE XYZ coordinates: Red: (0.4303, 0.2219, 0.0202), Green: (0.3416, 0.7068, 0.1296), Blue: (0.1784, 0.0713, 0.9393).
Smpte	SMPTE primaries with the following CIE XYZ coordinates: Red: (0.393, 0.212, 0.019), Green: (0.365, 0.701, 0.112), Blue: (0.192, 0.087, 0.958).

Remarks

The definition of the RGB primaries is not unique. In principle, there is one RGB system for each set of phosphors used in color monitors. Anyway, the CCIR has defined standard combinations for use in digital TV broadcast. Before performing a conversion, function [EasyColor::RgbStandard](#) can be used to specify the standard used.

6.96. ERoiHit Enum

Describes the ROI that was hit by the mouse cursor.

Namespace: Euresys.Open_eVision_2_10

NoHit	No ROI.
Learn_0	First learning ROI.
Learn_1	Second learning ROI.
Match_0	First matching ROI.
Match_1	Second matching ROI.

Inspect

Inspection ROI.

6-

.-

97. ESegmentationMethod Enum

The segmentation methods that are available to the image encoder.

Namespace: Euresys.Open_eVision_2_10

Custom	Custom local search neighborhood. Set EPatternFinder::AngleSearchExtent , EPatternFinder::ScaleSearchExtent , EPatternFinder::XSearchExtent and EPatternFinder::YSearchExtent to custom values.
BinaryImage	Segmentation of binary images (cf. EBinaryImageSegmenter).
ColorRangeThreshold	Segments a color image by specifying a cube in the RGB space (cf. EColorRangeThresholdSegmenter).
ColorSingleThreshold	Segments a color image by specifying a single threshold (cf. EColorSingleThresholdSegmenter).
GrayscaleDoubleThreshold	Segments a grayscale image by specifying a double threshold (cf. EGrayscaleDoubleThresholdSegmenter).
GrayscaleSingleThreshold	Segments a grayscale image by specifying a single threshold (cf. EGrayscaleSingleThresholdSegmenter).
ImageRange	Segments an image by specifying a pixel-by-pixel double threshold (cf. EGrayscaleDoubleThresholdSegmenter).

ReferencelImage	Segments an image by specifying a pixel-by-pixel single threshold (cf. EGrayscaleSingleThresholdSegmenter).
LabeledImage	Segments an image by mapping the value of the pixels directly to a layer index (cf. ELabeledImageSegmenter).

Remarks

The parameters of the segmentation methods are configured through the getters finishing by "Segmenter" that are available in [EImageEncoder](#).

6.98. ESegmentationMode Enum

Allowed values for the segmentation mode in EasyOCR.

Namespace: Euresys.Open_eVision_2_10

KeepObjects	After segmentation, keep the blobs as they were found.
RepasteObjects	After segmentation, group together the blobs believed to belong to the same character.

6.99. ESelectByPosition Enum

Allowed values for the selection mode of [ECodedImage](#).

Namespace: Euresys.Open_eVision_2_10

InsertIn	Insert the objects completely inside the given area.
InsertTouch	Insert all the objects with a non-empty intersection with the given area.

InsertOut	Insert the objects completely outside the given area.
RemoveIn	Remove the objects completely inside the given area.
RemoveTouch	Remove all the objects with a non-empty intersection with the given area.
RemoveOut	Remove the objects completely outside the given area.
RemoveBorder	Remove the objects outside the given area (including the objects touching the given area boundary).

Remarks

When specifying the position by means of an ROI, the minimum width and height of the ROI object must be at least 3 pixels. This enumeration pertains to the EasyObject legacy API and should not be used for new developments. See [ECodedImage2](#) for the new API.

6.100. ESelectionFlag Enum

Specifies to which subset of a selection an operation should be applied in EasyObject and EasyOCV.

Namespace: Euresys.Open_eVision_2_10

Any	The operation applies to both selected and unselected items.
True	The operation applies to selected items only.
False	The operation applies to unselected items only.

6.101. ESelectOption Enum

Allowed values for the selection mode of [ECodedImage](#). This enumeration pertains to the EasyObject legacy API and should not be used for new developments. See [ECodedImage2](#) for the new API.

Namespace: Euresys.Open_eVision_2_10

InsertAll	Add all objects.
InsertGreaterOrEqual	Add all objects with feature value above the upper threshold.
InsertLesserOrEqual	Add all objects with feature value below the lower threshold.
InsertRange	Add all objects with feature value between or equal to the lower and upper thresholds.
RemoveAll	Delete all objects.
RemoveGreaterOrEqual	Delete all objects with feature value above the lower threshold.
RemoveLesserOrEqual	Delete all objects with feature value below the upper threshold.
RemoveRange	Delete all objects with feature value between or equal to the lower and upper thresholds.
InsertOutOfRange	Add all objects with feature value above the upper and below the lower threshold.
RemoveOutOfRange	Delete all objects with feature value above the upper and below the lower threshold.

6.102. ESerializerFileWriterMode Enum

Creation mode of the file.

Namespace: Euresys.Open_eVision_2_10

Create	Creates the archive file on the hard disk. If the file already exists, the ESerializer::CreateFileWriter method returns NULL .
Overwrite	Overwrites the previously created archive if it already exists, or creates it otherwise.
Append	Appends the data at the end of the previously created archive, or creates the archive if it doesn't exist.

6.103. EShapeBehavior Enum

Allowed values for conditions on the behavior of a shape.

Namespace: Euresys.Open_eVision_2_10

Visible	Identifies a visible shape.
Selected	Identifies a selected shape.
Selectable	Identifies a selectable shape.
Dragable	Identifies a dragable shape.
Rotatable	Identifies a rotatable shape.

Resizable	Identifies a resizable shape.
Labeled	Identifies a labeled shape.
Active	Identifies an active shape.
Passed	Identifies a non defective shape.

6.104. EShapeType Enum

Gauge type.

Namespace: Euresys.Open_eVision_2_10

NoShape	-
FrameShape	Defines a frame shape.
WorldShape	-
PointGauge	Defines a point location gauge.
LineGauge	Defines a line fitting gauge.
CircleGauge	Defines a circle fitting gauge.
RectangleGauge	Defines a rectangle fitting gauge.
WedgeGauge	Defines a wedge fitting gauge.

6.105. EShiftingMode Enum

Allowed values for the shifting mode of EasyOCR.

Namespace: Euresys.Open_eVision_2_10

Chars	Each character is moved individually.
Text	The all set of characters is moved together.

6.106. ESingleThresholdMode Enum

The single threshold mode for the selection of coded elements with respect to a given feature.

Namespace: Euresys.Open_eVision_2_10

Less	The value of the feature must be strictly less than the threshold.
LessEqual	The value of the feature must be less or equal to the threshold.
Equal	Equality comparison. (*)
GreaterEqual	The value of the feature must be greater or equal to the threshold.
Greater	"Greater" comparison. (*)
Different	The value of the feature must be different to the threshold.

6.107. ESortDirection Enum

The sorting mode for selections of coded elements based on their features.

Namespace: Euresys.Open_eVision_2_10

Ascending	Sort by increasing feature values.
Descending	Sort by decreasing feature values.

6.108. ESortOption Enum

Allowed values for the sort mode of [ECodedImage](#). This enumeration pertains to the EasyObject legacy API and should not be used for new developments. See [ECodedImage2](#) for the new API.

Namespace: Euresys.Open_eVision_2_10

Ascending	Sort by increasing feature values.
Descending	Sort by decreasing feature values.

6.109. EStockMeasurementUnit Enum

Allowed values for the type of Measurement Unit.

Namespace: Euresys.Open_eVision_2_10

None	The object cannot be edited.

um	Defines the micron meter value type.
mm	Defines the millimeter value type.
cm	Defines the centimeter value type.
dm	Defines the decimeter value type.
m	Defines the meter value type.
dam	Defines the decameter value type.
hm	Defines the Hectometer value type.
km	Defines the Kilometer value type.
mil	Defines the milliliter value type.
inch	Defines the inch value type.
foot	Defines the foot value type.
yard	Defines the yard value type.
mile	Defines the mile value type.

6.110. ESymbologies Enum

The symbologies supported by EasyBarCode.

Namespace: Euresys.Open_eVision_2_10

Standard_Symbologies	Reserved for internal use
Additional_Symbologies	Reserved for internal use
Codabar	Codabar symbology.
Code128	Code 128 symbology.
Code25Interleaved	Code 25 Interleaved symbology.
Code39	Code 39 symbology.
Ean128	EAN 128 symbology.
Ean13	EAN 13 symbology.
Msi	MSI symbology.
UpcA	UPC A symbology.
UpcE	UPC E symbology.
BinaryCode	Binary Code symbology.
AdsAnker	Code ABC Anker symbology.
Bc412	Code BC 412 symbology.
Code11	Code 11 symbology.
Code13	Code 13 symbology.
Code25Datalogic	Code 25 DataLogic symbology.

Code25Matrix	Code 25 Matrix symbology.
Code25Iata	Code 25 IATA symbology.
Code25Industry	Code 25 Industry symbology.
Code25Compressed	Code 25 Compressed symbology.
Code25Inverted	Code 25 Inverted symbology.
Code32	Code 32 symbology.
Code39Extended	Code 39 Extended symbology.
Code39Reduced	Code 39 Reduced symbology.
Code93	Code 93 symbology.
Code93Extended	Code 93 Extended symbology.
CodeBcdMatrix	Code BCD Matrix symbology.
CodeCip	Code CIP symbology.
CodeStk	Code STK symbology.
Ean8	EAN 8 symbology.
IbmDeltaDistanceA	IBM Delta Distance A symbology.
Plessey	Plessey symbology.
Telepen	Telepen symbology.

Rss14	RSS-14 symbology.
Rss14Limited	RSS-14 Limited symbology.
Rss14Expanded	RSS-14 Expanded symbology.
Standard	Allows positioning the shape edges symmetrically.
Additional	Gathers all the symbologies belonging to the additional group.
Unknown	Calibration type is not defined.

Remarks

Due to the large number of supported symbologies, they have been splitted into two groups. The most commonly used symbologies have been gathered under the name Standard symbologies. The remaining symbologies belong to the Additional symbologies group.

6.111. EThinStructureMode Enum

Allowed values for the type of thin structures in EasyFind.

Namespace: Euresys.Open_eVision_2_10

Auto	Chooses automatically the orientation of the ZMap.
Dark	Favors thin elements darker than regions.
Bright	Favors thin elements brighter than regions.

6.112. EThresholdMode Enum

The various modes for thresholding that are supported by Open eVision.

Namespace: Euresys.Open_eVision_2_10

Absolute	The absolute value is used.
Relative	Relative threshold; determines the required threshold level so that a given fraction of the image pixels lie below it.
MinResidue	Selects a threshold value such that the quadratic difference between the source and thresholded image is minimized.
MaxEntropy	Selects a threshold value such that the entropy (i.e. the amount of information) of the resulting thresholded image is maximized.
Isodata	Selects a threshold value that lies halfway between the average dark gray value (i.e. gray levels below the threshold) and average light gray values (i.e. gray levels above the threshold).

6.113. ETransitionChoice Enum

The transition selection method applied by the gauge measurement

Namespace: Euresys.Open_eVision_2_10

--

NthFromBegin	N-th transition from the beginning (counting from 0).
NthFromEnd	N-th transition from the end (counting from 0).
LargestAmplitude	Transition whose peak has the largest amplitude value.
LargestArea	Transition whose peak has the largest area value.
Closest	Transition closest to the center.
All	All transitions.

6.114. ETransitionType Enum

The type of transition to be retained by the gauge measurement

Namespace: Euresys.Open_eVision_2_10

Bw	Black to white.
Wb	White to black.
BwOrWb	Black to white or white to black.
Bwb	Black to white to black.
Wbw	White to black to white.

6.115. EZMapOrientationVectorMode Enum

The [EZMap](#) orientation, it's the direction of the X (width) axis of the ZMap.

Namespace: Euresys.Open_eVision_2_10.Easy3D

Auto	Chooses automatically the orientation of the ZMap.
XAxis	The X (width) axis of the ZMap is aligned with the world X axis.
YAxis	The X (width) axis of the ZMap is aligned with the world Y axis.
ZAxis	The X (width) axis of the ZMap is aligned with the world Z axis.
Explicit	The reference plane is arbitrary and given by the method <code>SetReferencePlane</code> .

6.116. EZMapReferencePlaneMode Enum

The 3D reference plane used to build the [EZMap](#).

Namespace: Euresys.Open_eVision_2_10.Easy3D

XPlane	The reference plane is orthogonal to the X axis (YZ domain).
YPlane	The reference plane is orthogonal to the Y axis (XZ domain).
ZPlane	The reference plane is orthogonal to the Z axis (XY domain). That's the default reference plane.
Explicit	

6.117. Features Enum

The reference plane is arbitrary and given by the method `SetReferencePlane`.

Open eVision Features.

Namespace: Euresys.Open_eVision_2_10.LicenseFeatures

EasyGauge	-
EasyColor	-
EasyImage	-
EasyObject	-
EasyBarCode	-
EasyMatch	-
eVisionStudio	-
EasyFind	-
EasyMatrixCode	-
EasyOCR	-
EasyOCV	-
EasyQRCode	-

EasyOCR2	-
Easy3D	-
EasyDeepLearning	-
Easy3DObject	-
Easy3DMatch	-
Easy3DLaserLine	-