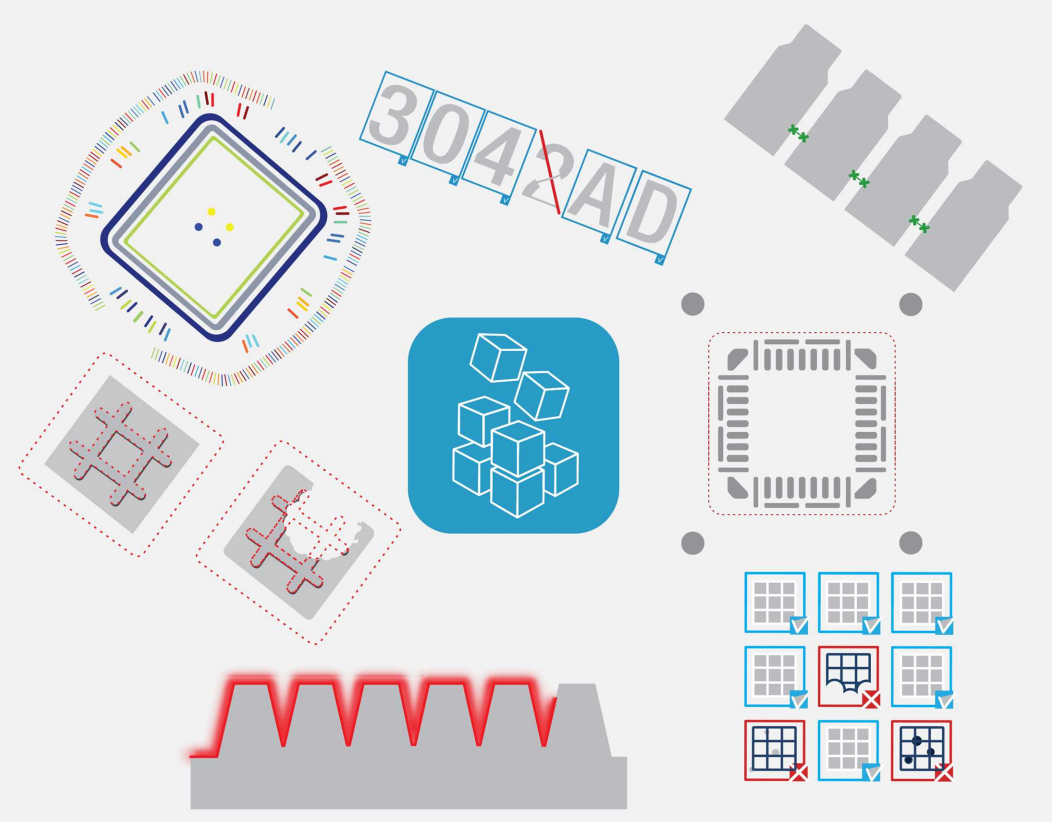
euresys
Empowering Computer Vision

RELEASE NOTES

Open eVision

Release 22.04.0



This documentation is provided with **Open eVision 22.04.0** (doc build **1166**).
www.euresys.com

Contents

1. Release Benefits 4
2. Release Specifications 6
3. End of Life Announcements 8
4. Release Details 9
   4.1. New and Improved Features 9
   4.2. Breaking Changes 15
   4.3. Changes 16
   4.4. Solved Issues 17
5. Known Issues 19

1. Release Benefits

Release version

Starting with this release, the **Open eVision** version number will follow a calendar based convention. The format will be: YY.MM.RR.BB.

- YY - year of the release
- MM - month (with 0 padding)
- RR - revision index
- BB - build number

Release benefits

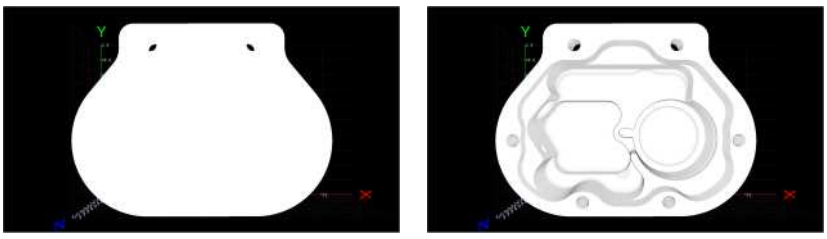
Starting with this release **22.04**, **Open eVision** offers the following new products and functions:

Embedded systems

- **Open eVision** for embedded systems is now available. It supports the ARMv8-A 64-bit architecture (like Cortex-A53 or Cortex-A72 cores).
- All **Open eVision** features are supported, with the following exceptions:
  - For **Easy3D** library: the `E3DViewer` class is not available.
  - For **EasyClassify**, **EasySegment** and **EasyLocate**:
    - GPU accelerated training and inference are not available yet, but they will be available in the next release.
    - CPU-based inference and training are available.
  - For licensing: only the **Neo Licensing System** is supported (local software container or USB dongle container).

Easy3D

- The 3D viewer can now shade opaque rendering sources with the Eye-Dome-Lighting (EDL) algorithm.
  - To enable or disable it, use `E3DViewer.SetEnableEDLShading`
  - It is enabled by default if the hardware is compatible (the minimal OpenGL version is 3).

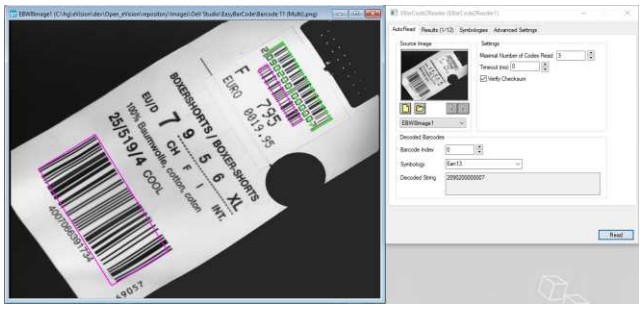


Without EDL (left) and with EDL (right)

EasyBarCode

- **EasyBarCode2** now supports the **Pharmacode One Track** symbology. The C++ and C# BarCode2 code samples have been updated so you can choose the symbologies to detect.

- **EasyBarCode2** is now available in **Open eVision Studio**. To try it out, click on `EasyBarCode` > `New BarCode2 Tool`.



2. Release Specifications

OS and processor architectures

Windows OS

- **Open eVision** is a 32-bit and 64-bit library that requires a processor compatible with the SSE4 instruction set.
- **Open eVision** runs on the following Windows operating systems:

OS version	Additional information	
Windows 11®	64-bit	—
Windows 10®	32-bit	—
Windows 10®	64-bit	—
Windows 8®	32-bit	—
Windows 8®	64-bit	—
Windows 7®	32-bit	The recommended version is 6.1.7601 (Windows 7 Service Pack 1)
Windows 7®	64-bit	

Linux OS

- **Open eVision** is compatible with x86_64 and aarch64 (ARM) CPUs.
- **Open eVision** and the **Neo License Manager** are designed to be distribution-independent on x86_64 platforms.
- **Open eVision** is expected to work with all `deb`/rpm based distributions with `glibc` version 2.17 or newer.
- This release has been validated with the following distributions and their default `gcc` compilers and `cmake` programs:
  - **Ubuntu LTS** 16.04 to 20.04
  - **CentOS** 7 and 8
  - **Fedora** 33 to 35
- This release has been validated on the following embedded systems:
  - **Raspberry Pi** 4 and Raspberry Pi Zero 2 W
  - **NVidia** Jetson Nano & Xavier AGX
  - Smart camera **ADLink** NEON-2000-JNX

Qt compatibility

Some **Open eVision** samples use the **Qt** framework to create a user interface and display images with a graphical overlay.

- We recommend the **Qt** versions 5.12 to 5.15.

Remote connections and virtual machines

- Remote connections
  - You can install and use **Open eVision** licenses on a remote connection using remote desktop, TeamViewer or any other similar software.
- Virtual machines
  - Linux virtual machines are supported. Microsoft Hyper-V and Oracle VirtualBox hypervisors have been successfully tested.
  - You cannot install **Open eVision** on Windows virtual machines.

Supported IDE and programming languages

Select the recommended API Module according to your IDE and programming language:

	Programming language	
IDE	**C++**	**C#, VB.NET, C++/CLI**
Microsoft Visual Studio 2008® SP1	C++	.NET Assembly
Microsoft Visual Studio 2010®	C++	.NET Assembly
Microsoft Visual Studio 2012®	C++	.NET Assembly
Microsoft Visual Studio 2013® (*)	C++	.NET Assembly
Microsoft Visual Studio 2015®	C++	.NET Assembly
Microsoft Visual Studio 2017®	C++	.NET Assembly
Microsoft Visual Studio 2019®	C++	.NET Assembly
Microsoft Visual Studio 2022®	C++	.NET Assembly
QtCreator 4.15 with Qt 5.12 (**)	C++	

NOTE
(*) **Visual C++ MFC MBCS Library for Visual Studio 2013** must be installed.
(**) A C++ compiler like **MSVC** must be installed

Required system resources

- Processor:
  - Intel compatible processor supporting the SSE4 instruction set
  - ARMv8-A compatible processor
- Memory:
  - Execution: minimum 2GB RAM to run an **Open eVision** application
  - Compilation: minimum 8GB RAM to compile an **Open eVision** application
- Hard disk space:
  - **Open eVision** libraries: 100 MB - 2 GB (depending on selected options)

3. End of Life Announcements

Legacy Headers

- The **Legacy Headers** will be removed from **Open eVision** distributions at the end of 2022.
  - The **Legacy Headers** help the customers to migrate an existing application using **eVision** (last major release in 2006) to **Open eVision**.
  - The 2022 releases of **Open eVision** will be the last ones including these **Legacy Headers**.

Visual Studio 2008

- The support for **Visual Studio 2008** will stop at the end of 2022.
  - The minimum **Windows** development environment will then be **Visual Studio 2013**.

32-bit libraries

- The **32-bit Open eVision** libraries (only available for **Windows**) will be removed from **Open eVision** distributions in July 2023.
  - You should migrate to 64-bit if you need newer versions of **Open eVision**.

4. Release Details

4.1. New and Improved Features

New features

Embedded systems

- **Open eVision** for embedded systems is now available. It supports the ARMv8-A 64-bit architecture (like Cortex-A53 or Cortex-A72 cores).
- All **Open eVision** features are supported, with the following exceptions:
  - For **Easy3D** library: the `E3DViewer` class is not available.
  - For **EasyClassify**, **EasySegment** and **EasyLocate**:
    - GPU accelerated training and inference are not available yet, but they will be available in the next release.
    - CPU-based inference and training are available.
  - For licensing: only the **Neo Licensing System** is supported (local software container or USB dongle container).
- **Open eVision** for embedded systems has been tested on Raspberry PI (4 and Zero 2), NVidia Jetson (Nano and Xavier AGX) and ADLink Neon-2000 smart camera.
- Sample programs and other resources for embedded systems like Raspberry PI, NVidia Jetson or ADLink Neon smart camera are provided in an Additionnal Resources package available on the website download area.

Neo Licensing System

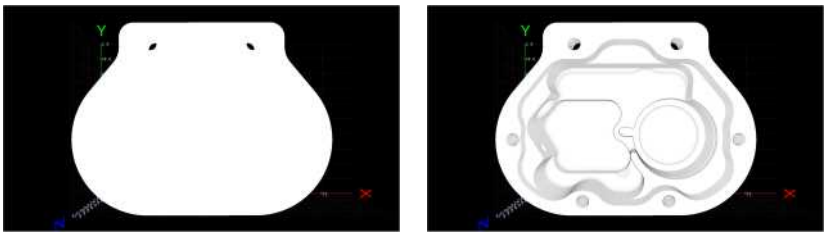
- Use the new `Easy::SetOemKey` and `Easy::CheckOemKey` overloads that use a key index to store up to 12 independent OEM keys in a single dongle.

EasyImage

- `EasyImage::Median` now supports median filters of arbitrary size (except when image type is an `EROIBW►`).

Easy3D

- The 3D viewer can now shade opaque rendering sources with the Eye-Dome-Lighting (EDL) algorithm.
  - To enable or disable it, use `E3DViewer.SetEnableEDLShading`
  - It is enabled by default if the hardware is compatible (the minimal OpenGL version is 3).



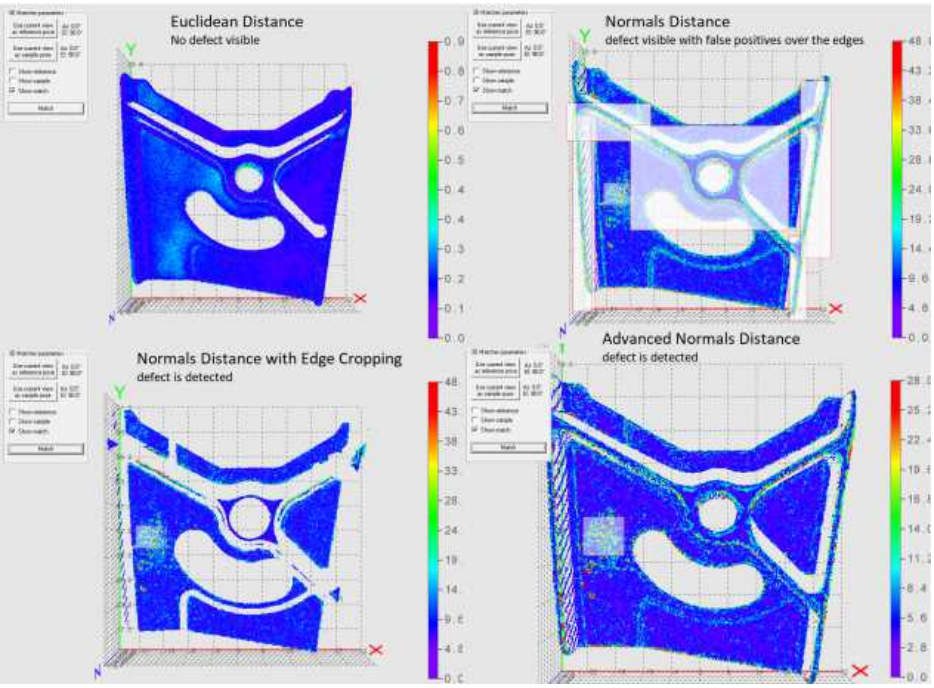
Without EDL (left) and with EDL (right)

- Use the new method `EFilters::Median` to perform median filtering of arbitrary size on an `EZMap` or an `EDepthMap`. The filter can ignore undefined pixels or not (it is faster to not ignore them).
- Use the new class `EPointCloudFilter` to filter out noise from point clouds, in a similar fashion to what `EFilters::RemoveNoise` do on `EZMap` or `EDepthMap`.
- The 3D viewer can now display planes, lines, boxes and spheres. You can also choose the color and the opacity.
- The new class `E3DLine` is available.
- The facet normals are now supported in `E3DMesh`. The normals are read from `.stl` files and can be retrieved from the `EMesh` object.
- Use the class `EZMapToPointCloudConverter` to generate normals that are stored as an attribute of the `EPointCloud`.
- Use the new method `EPointCloud::ComputeNormalsAndCurvatures` to compute the normal (and curvature) for each point based on its neighbors.
- In `EPointCloud`, the new attribute `E3DAttribute_Curvature` stores the curvatures.
- In `E3DViewer`, use `EColorRampMode_HueFromNormal`, the new value for the enum `EColorRampMode`, to compute the colors of the `EPointCloud` or of the `EMesh` from its normals.

Easy3DMatch

- `E3DMatcher` and `E3DComparer` now support the comparison based on the normals of the model and reference.
  - There are 2 new values for `EComparisonDistanceMode`
  - The existing values are renamed for improved clarity.

  These comparison modes are well suited to detect scratches as illustrated below.



- `E3DMatcher` and `E3DComparer` can now perform a comparison without taking the edges of the object into account (`EnableAutomaticEdgeCropping`). This is especially interesting when using comparison based on normals as normals are badly defined around edges.

EasyMatrixCode2

- A new option allows the reader not to be strict during decoding to be able to read codes with some small encoding errors.

EasyOCR2

- To recognize specific symbols, use different classifiers instead of the classifier set by `EOCR2.SetClassifier`, with the new methods `EOCR2.AddClassifierForSymbol`, `EOCR2.GetClassifierForSymbol` and `EOCR2.RemoveClassifierForSymbol`.

  Currently, you can use only `EOCR2Classifier_DatabaseClassifier` with these methods.

Samples

- Some of our sample programs were available only in C++.

  C# samples are now available for `Easy3DPhotometricStereo`, `Easy3DViewer`, `Easy3DMatchPointCloudMerger`, `EasyClassifyTrain`, `EasyLocateTrain`, `EasySegmentSupervisedTrain`, `EasySegmentUnsupervisedTrain`, `EasyObjectEChecker2` and `EasyMatrixCodeRead`.

- To improve readability, the sample programs are now organized into the following 5 folders:
  - 3D Processing
  - Deep Learning Inspection
  - Image Processing
  - Matching and Measurement
  - Text and Code Reading
- The samples are also renamed following the pattern "LicenseFunction", for example: EasyFindLearn or EasyImageFilter.

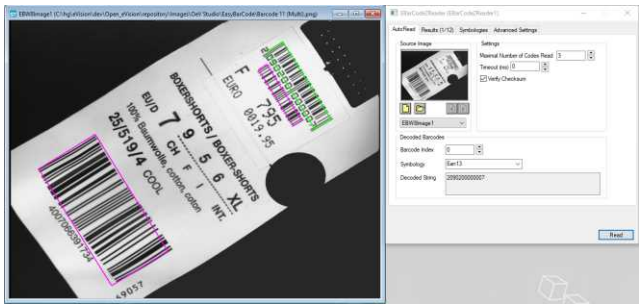
EasyDeepLearning

- In the API, use the new method EDeepLearningTool::InitializeInference to launch various initializations that are usually done during the first inference performed by the tool. Use this method to get a consistent inference time for your first inference that was much slower than the following ones.

EasyBarCode

- **EasyBarCode2** now supports the **Pharmacode One Track** symbology. The C++ and C# BarCode2 code samples have been updated so you can choose the symbologies to detect.

- **EasyBarCode2** is now available in **Open eVision Studio**. To try it out, click on EasyBarCode > New BarCode2 Tool.



- The BrcReading Qt sample is updated and uses **EasyBarCode2** instead of **EasyBarCode1** so you can choose the symbologies to detect.
- In **EasyBarCode2**, use the new method EBarCodeReader::SetMinModuleSize to specify the size of the smallest module. While not required in most cases, this can help to read small barcodes in large images.

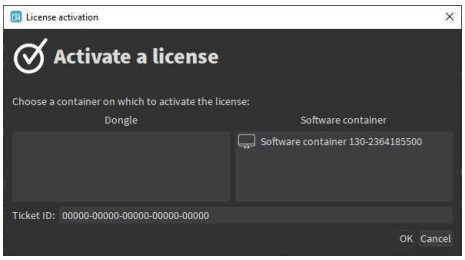
Improvements

EasyMatrixCode2

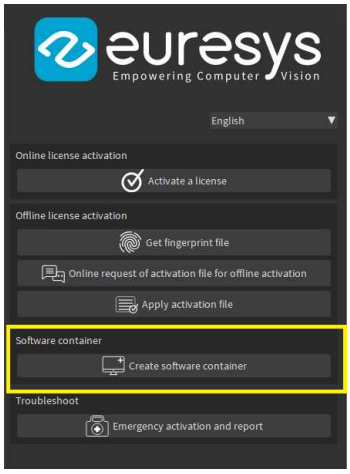
- The reading of codes with a logical height to width ratio larger or equal to 4 is improved.
- The overall decoding speed is improved.
- The rejection of false positive detections is better.

Neo License Manager

- The graphical user interface of the **Neo License Manager** is improved.
  - The dongle and software containers are now clearly differentiated in the license activation dialogs.

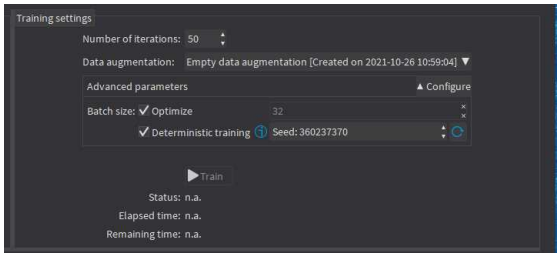


  - The action to create a software container is now only available as a button on the main window of the **Neo License Manager**.



Deep Learning Studio

- A splashscreen is now displayed while **Deep Learning Studio** is loading.
- You can now control the random seed used to perform a deterministic training in **Deep Learning Studio**.



- In the tool tab, you can select and remove several tools at the same time. A confirmation dialog is displayed if one of the tools is trained.

4.2. Breaking Changes

Starting with this release **22.04**, **Open eVision** implements the following changes:

Easy3D

- The classes `EPointCloud::FillAttributeBuffer` and `EPointCloud::AddCustomAttributeBuffer` are updated to use vectors instead of pointers so the C# API is functional. breaking change
  - The code snippets illustrating how to use these functions in C# and C++ are added.

4.3. Changes

Starting with this release **22.04**, **Open eVision** implements the following changes:

Easy3D

- *(Breaking change)* The classes `EPointCloud::FillAttributeBuffer` and `EPointCloud::AddCustomAttributeBuffer` are updated to use vectors instead of pointers so the C# API is functional. breaking change
  - The code snippets illustrating how to use these functions in C# and C++ are added.

Easy3DObject

- `E3DObjectExtractor` now correctly takes into account user-defined min and max limits for the length and the width of extracted objects.
  - The search excludes objects bordering on the limits of the image.
  - The search excludes a region of interest that prevents partial object detection and the detection of the whole region as an object.

EasyImage

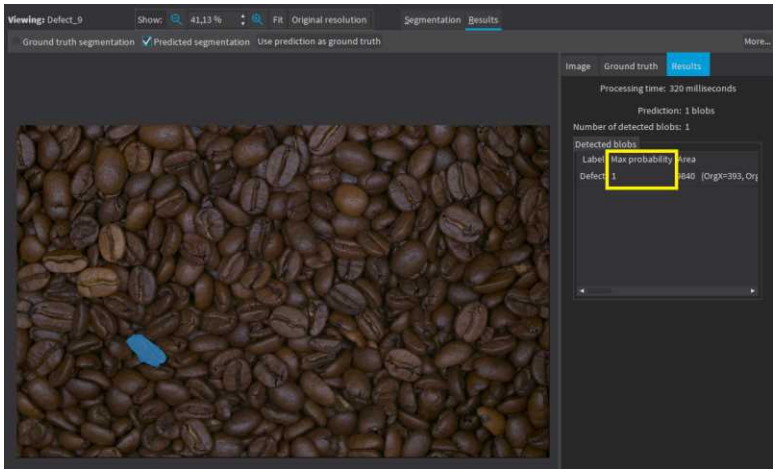
- All the **EasyImage** functions now check that the ROI they use as parameters are valid. More precisely, **Open eVision** checks that the ROI does not point outside of its parent image.

EasyBarCode

- In **EasyBarCode2**, the error correction is slightly improved for symbologies having a well-defined checksum (ean8/13/128, code 128, code 93, UPCA and UPCE).

Deep Learning Studio

- For **EasySegment Supervised**, the maximum probability is displayed for each instead of the average probability. This is because the classification threshold is applied on the maximum probability and not on the average probability.



4.4. Solved Issues

The following issues have been fixed in **Open eVision 22.04**:

Easy3D

- *(22.04.0)* Sometimes, the normals computed when applying an `E3DTransformMatrix` to an `EPointCloud` having normals had the wrong direction. This has been fixed.
- *(22.04.0)* The class `EMesh::LoadSTL` now correctly reads binary `.stl` files from **Solidworks**.

EasyOCV

- *(22.04.0)* The method `EOCV::DrawTemplateChars` now correctly displays the template characters.

EasyLocate

- *(22.04.0)* **EasyLocate** could find one object more than the specified maximum number of objects property. This has been fixed.

Easy3DObject

- *(22.04.0)* The base plane extraction depended on the resolution of the source image. From now on, the extraction is normalized.

EasyImage

- *(22.04.0)* Using the operation `EArithmeticLogicOperation_ShiftLeft` or `EArithmeticLogicOperation_ShiftRight` with a shift higher than the maximum number of bits representing the data now throw an `EException`. Previously the behavior was undefined.
- *(22.04.0)* The conversion from color to gray was incorrect. This has been fixed. It also takes into account the current `ERgbStandard`.
- *(22.04.0)* Loading a color image into `EImageBW8` now takes into account the current `ERgbStandard`.
- *(22.04.0)* Contour now correctly returns the path instead of throwing an exception if the object touches the image border.

EasyBarCode

- *(22.04.0)* **EasyBarCode2** did not eliminate the bar codes with wrong checksums when `EBarCodeReader::GetValidateWithChecksum` was true. This has been fixed.

  As a consequence, you may need to disable the checksum validation in your application.

EasyFind

- *(22.04.0)* A limitation that appears when using many parallel threads (20+) with **EasyFind** on multiple CPU system has been fixed. That limitation could cause **EasyFind** to slow down after some time.
- *(22.04.0)* You can now save the pivot parameter of `EPatternFinder`.

EasyMatch

- *(22.04.0)* When matching a very large pattern (more than 5000 × 5000 px), the results could have a score of 0 instead of the correct value. This has been fixed.

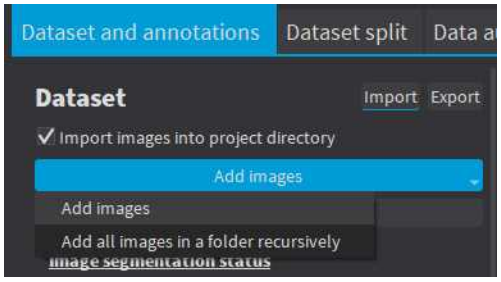
  As a consequence, the computed scores can now very slightly vary in some cases (we observed changes of around 5e-7).

EasyQRCode

- *(22.04.0)* An issue where an unhandled exception could be thrown has been fixed.

Deep Learning Studio

- *(22.04.0)* The `Add all images in a folder recursively` button has been fixed.



- *(22.04.0)* Various display problem when editing the ground truth segmentation of an image have been fixed.

5. Known Issues

Samples

- *(22.04.0)* In the Qt sample `Easy3DMatchAlign`, on **Ubuntu 16.04** and with **Qt version 5.5.1**, it is not possible to rotate or translate the 3D view.

Licensing

On some installations, the licensing systems can take a long time to start (from 10 seconds up to a few minutes). If you have this issue, you can try the following procedures:

- Clean your software license cache.
  - The software license cache can become bloated by usage.
  - It can also happen if you use only dongles, as the system checks the presence of software licenses in all cases.
  - To clean the cache, use the `LicenseManager.exe /DeleteLicenseFiles` command.

> **WARNING**
> This command deletes all the licenses that are not managed by the **Neo License Manager** on the system. Reactivate these licenses after the cleaning.

- Update your system root certificates.
  - If your root certificates are expired, the validation of the licensing system signatures might fail and timeout.
  - This only happens if the computer is on a network, even if the network is not connected to the Internet.
- Enable only the licensing system(s) you use.
  - By default, all the supported licensing systems are enabled.
  - Use the new (available from 2.13) `Preconfiguration::SelectLicensingModels` method to select exactly the licenses you want to enable and avoid issues arising from the usage of the other ones.

.NET API and unsigned integer parameters

Since this release 2.5 of **Open eVision**, unsigned integer parameters in the C++ API are not exposed in the .NET API as signed integer anymore, but as unsigned integers. This brings the .NET API closer to the C++ one.

This change does not cause any issue except when you want to pass an enumerate value as one of these parameters. In these specific cases, update your casting operation as in the following example:

```
codedImage.SetThreshold((int)EThresholdMode.MinResidue);
```

becomes:

```
codedImage.SetThreshold(unchecked((uint)EThresholdMode.MinResidue)); .
```

Reserved keywords

- The following keywords are reserved by **Open eVision**:
  - EUnit_um, EUnit_mm, EUnit_cm, EUnit_dm
  - EUnit_m, EUnit_dam, EUnit_hm, EUnit_km
  - EUnit_mil, EUnit_inch, EUnit_foot, EUnit_yard, EUnit_mile
  - EasyWorld

TIP
To avoid conflict, do not use these keywords to name variables, functions, methods, macros...

Image formats

- If you use some types of 96-bit RGB Tiff image, **Open eVision** may crash.

Memory leaks

- If you use the CRT library to detect memory leaks in your program, it can falsely detect some memory leaks when you use the **Open eVision** library.
  - This is a known limitation of the CRT library memory leak detection scheme.
    See: `https://docs.microsoft.com/en-us/visualstudio/debugger/ finding-memory-leaks-using-the-crt-library`
  - It happens when the memory leak detection scheme is ended before the **Open eVision** DLL is unloaded or the code in the **Open eVision** headers is uninitialized.

Object cleanup: .NET

As a rule, it is highly recommended to call `Dispose()` on **Open eVision** .NET objects when they are not useful anymore.

TIP
Not doing so might result in unnecessarily high memory usage and crashes.

Example in C#

```
using(EImageBW8 src = new EImageBW8())
using(EPatternFinder finder = new EPatternFinder())
{
  src.Load(ImageFilePath);
  EFoundPattern[] foundPatterns = finder.Find(src);
  ...
  foreach(EFoundPattern foundPattern in foundPatterns)
  {
    foundPattern.Dispose();
  }
}
```

In addition, if you use a nested object (such as the segmenter properties in EasyObject encoder objects), remember to call `Dispose()` on that object before calling `Dispose()` on the parent object.

Example in C#

```
imageEncoder.GrayscaleSingleThresholdSegmenter.BlackLayerEncoded = true;
...
imageEncoder.GrayscaleSingleThresholdSegmenter.Dispose();
imageEncoder.Dispose();
```

Basic types: retrieving and setting pixel values

Using the `GetPixel()` and `SetPixel()` methods of the various ROI classes can sometimes be slow if you make many calls (regardless of the language used).

- In order to greatly speed up the ROI/image buffer access, embed the buffer access in your own code.
- See the examples below that use the new **Open eVision** API.

> **NOTE**
> For a better readability of these examples, the variable declarations and initializations have been omitted when possible.

Example in C++

```
void* pixAddr;
UINT8 pix;
...
for (int y = 0; y < height; ++y)
{
  pixAddr = bw8Image.GetImagePtr(0,y);
  for (int x = 0; x < width; ++x)
  {
    pix = *(reinterpret_cast<UINT8*>(pixAddr)+x);
  }
}
```

Example in C#

```
using System.RunTime.InteropServices;
...
IntPtr pixAddr;
byte pix;
...
for (int y = 0; y < height; ++y)
{
  pixAddr = bw8Image.GetImagePtr(0,y)
  for (int x = 0; x < width; ++x)
  {
    pix = Marshal.ReadByte(pixAddr,x)
  }
}
```

Basic types: ROI zooming and panning issue

- When drawing an ROI with a zoom factor, applying panning (retrieved from a scroll bar) causes the ROI display to be shifted. Consequently, the `HitTest()` and `Drag()` functions fail because the handles do not appear at their actual positions.

  Workaround: The panning values should be divided by the zoom factor before calling the `DrawFrame()`, `HitTest()` and `Drag()` functions.

Basic Types: miscellaneous issues

- TIFF files containing RGB values + alpha values are not supported.
- Filenames with multibyte characters are not supported. The error is "Unrecognized file format".
  - Use UTF-8 encoded strings to handle filenames with non-latin characters.
- `Easy::GetBestMatchingImageType()` only works for BW8 and C24 images.

Multithreading

- In multithread applications, if `Easy::Initialize` is not called before launching new threads that call **Open eVision** functions, then the number of **Open eVision** processing threads in these new threads may be wrongly initialized to use all the cores that are available on the machine.

EasyBarCode

- Due to a bug in the debugger of Visual C++ 2012, the reading time of bar codes may increase after a failed reading. This happens only in debug mode with Visual C++ 2012.
- EasyBarCode requires that a quiet zone of at least one full module is present around the whole bar code to be read.
- EasyBarCode is currently unable to read bar codes with curved or distorted bars. For reliable reading, the bars must be as straight as possible.
- EasyBarCode is currently not multithread-safe.

EasyOCR2

- *(2.13.0)* The detection of a topology with ranged characters was always failing with the proportionnal detection method. This functionnality is now disabled and an error is thrown.

EasyQRCode

- EasyQRCode does not support MicroQR codes.

EasyObject

- The `ECodedImage2` and `EHarrisDetector` results are drawn slowly when there are many results.

EasyMatch

- By design, the maximum size for a pattern in EasyMatch is 1791 x 1791.
- Matching a vertically symmetric pattern with an angle tolerance around 180° and in the original image can lead to an error of 1 pixel on the detected position.
- By default, EasyMatch interpolation does not work on 15 x 15 and smaller patterns.

  Workaround: For pattern sizes smaller than 16 x 16, adjust the `MinReduced` area to fit the `MinReducedArea < W*H/4` (if interpolation is needed).

EasyGauge

- In .NET, the `EPointGauge.GetMeasuredPoint()` overload with no argument is not available. To get the default measured point, use -1 as index.
- By design, an `ELineGauge`, `ERectangleGauge`, `ECircleGauge` or `EWedgeGauge` is reported as invalid if at least one of its sample points is invalid. In addition, these invalid sample points cannot be drawn as they have not been measured successfully.
- The `EWedgeGauge::SetActiveEdges()` method incorrectly gets the `EDragHandle_Edge_r` and `EDragHandle_Edge_RR` bits mixed up when processing its argument.

  Workaround: In order to activate the inner circle, set the `EDragHandle_Edge_RR` flag and use the `EDragHandle_Edge_r` flag to activate the outer circle.
- Using a gauge on an ROI leads to drawing problems.

  Workaround: Use the gauge on the parent image.
- In the custom `EDraggingMode_ToEdges` dragging mode, you cannot resize the nominal wedge gauge position using the on-screen handles, neither in a custom application nor in **Open eVision Studio** or in **Open eVision** Eval.

  Workaround: Enter numerical values for the wedge gauge position.

EasyMatrixCode

- When grading is enabled, the optimizations are made in order to get accurate grading rather than have the best possible reading. As a result, the number of decoding errors reported with grading can be higher than without grading.
- Inspecting images with a lot of details, even if they are low contrast, can require much more time spent in EasyMatrixCode than the `TimeOut` set previously.
- In .NET, retrieving the coordinates of a MatrixCode using `EMatrixCode.GetCorner()` or `EMatrixCode.Center()` can lead to an unhandled exception when the garbage collection starts up. To avoid this problem, call `Dispose()` on the `EPoint` objects returned by these functions when they are no longer needed.

Easy3DObject

- The `E3DObjectExtractor` objects saved with **Open eVision** 2.11 cannot be loaded with **Open eVision** 2.12.

Easy3D

- *(2.16.0)* When using the class `EMeshToZMapConverter` without extension, some triangles of the mesh close to the ZMap border may be removed.

EasyDeepLearning

- *(2.15.0)* The Deep Learning tool objects (`EClassifier`, `EUnsupervisedSegmenter`, `ESupervisedSegmenter` and `ELocator`) can leak CPU and GPU memory at destruction.

  So, it is not recommended to create and delete a lot of Deep Learning tools in the same program.

Open eVision Studio

- In the ROI management dialog, clicking on a ROI in the tree view does not activate the ROI overlay in the image window. This can prevent you to graphically interact with it.

  To avoid this issue and to properly interact with the ROI overlay:

  a. Click on the ROI in the tree view.

  b. Immediately after, click inside its overlay in the image window.
- To avoid crashes, deselecting all detection methods in the EasyQRCode dialog box reverts to the default detection method. In some cases, the dialog might not refresh automatically.
- In the detection method selection control of the EasyQRCode dialog box, clicking beside a text might select or deselect it.
- When managing the EasyOCR2 topology, the potential characters option is not available.

Open eVision installer

- There is a conflict between the **Open eVision** installer and any program using the UDP:6001 port. When a software is already using this port, the installation fails and rolls back.

  Workaround: Install **Open eVision** first, and then the other software.

> **NOTE**
> This port is typically used by National Instrument software such as LabView.

- Before installing any Euresys product, make sure that your OS is up-to-date (using Microsoft Update), otherwise, problems might occur.

Open eVision License Manager

- Under Windows XP, the **Open eVision License Manager** might not start if the .NET Framework 2.0 is not installed.
- Using the **Open eVision License Manager** to activate a license requires an Internet connection and a secure SSL transaction to **EURESYS S.A.** servers.

> **NOTE**
> On older systems, such as Windows XP SP3, ensure that the root certificates are up-to-date otherwise the secure connection is refused and the license is not activated.

- When activating an emergency license, the following error may occur: "`Error Message: Loading of the ASR failed!`"

  This error occurs when all 3 emergency licenses have already been used and the computer has been formatted.
- Using **Open eVision License Manager** in English language mode on a Chinese or Japanese Windows version can lead to truncated text being displayed. This is an issue linked to the automatic font selection and there is currently no workaround. Please note however that, by default, the **Open eVision License Manager** runs in the OS language, including Chinese and Japanese.