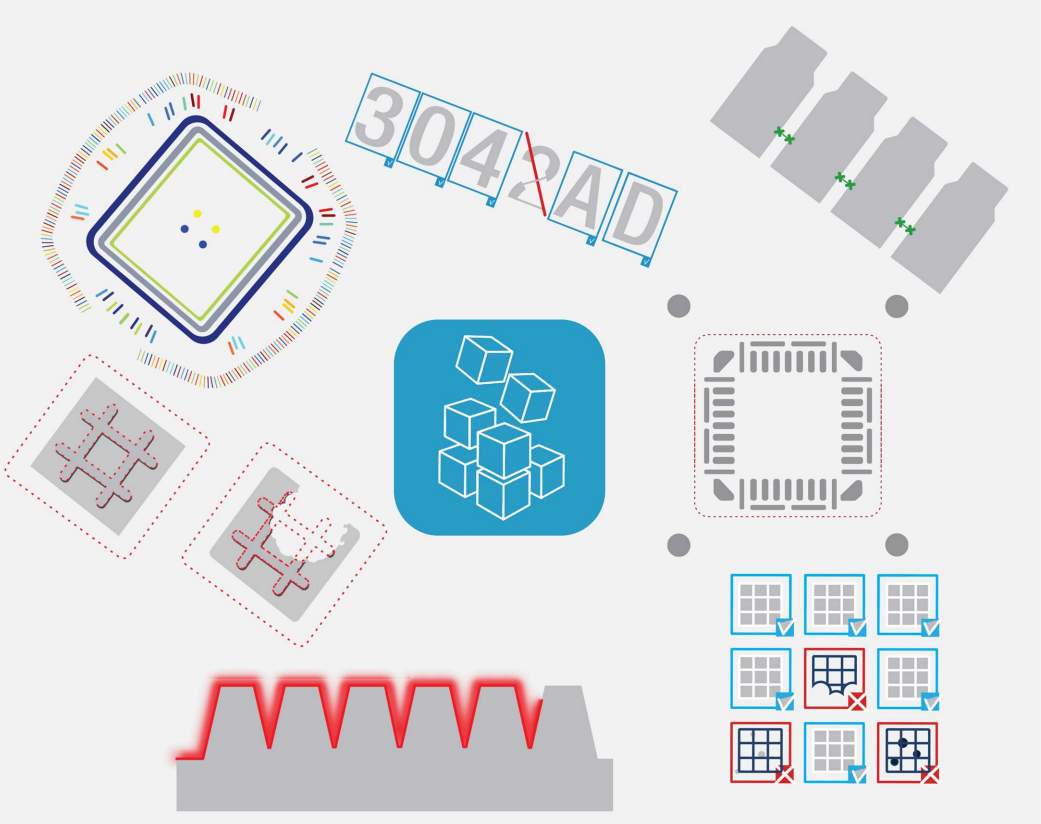
euresys
Empowering Computer Vision

RELEASE NOTES

# Open eVision

## Release 25.10.1



© EURESYS S.A. 2025 - Doc. D101EN-Release Notes-Open eVision-25.10.1.1224 built on 2025-11-24

This documentation is provided with **Open eVision 25.10.1** (doc build **1224**).
www.euresys.com

This documentation is subject to the General Terms and Conditions stated on the website of **EURESYS S.A.** and available on the webpage https://www.euresys.com/en/Menu-Legal/Terms-conditions. The article 10 (Limitations of Liability and Disclaimers) and article 12 (Intellectual Property Rights) are more specifically applicable.

# Contents

# 1. Release Benefits

What's new in **Open eVision 25.10**

## New reference frame and alignment object EFrame

The new EFrame represents a local reference frame and is used to express various data structures of **Open eVision** such as regions, geometrical primitives, gauges, EFoundPattern, bar codes, matrix codes, QR codes (and more in future releases).

- Use the method ToFrame of a data structure to convert it to a frame.
- You can calibrate the frame so it represents a reference frame in the real world.
- The old class EFrame is now part of the namespace Euresys::Open_eVision::Legacy.

## EasyImage

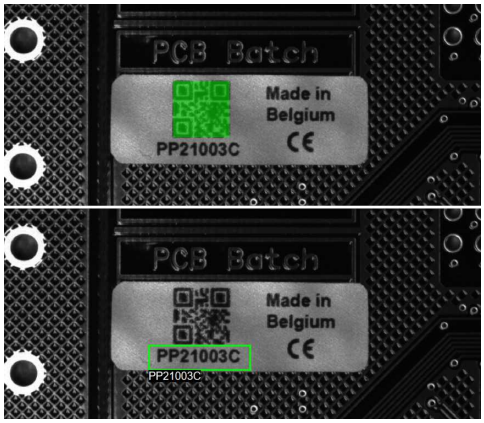
- EImageStitcher:
  - The new "Incremental Stitching" mode enables the overlapping of the image capture and the stitching operations to reduce the overall processing time while maintaining identical results.
  - In grid mode, the new parameters overlapX and overlapY specify the overlaps between adjacent images on the X and Y axes.
  - Use the new algorithm parameter EImageStitcher.Algorithm to favor speed or robustness.
  - Use the new Learn method to precompute the transformations to apply to the set of images given as input. After a successful Learn, the subsequent calls to the stitcher are faster and use the same transformations.
  - You can now give to the stitcher a vector of ERegion as an additional input. Each region of this vector indicates the portion of the image containing the original image after an Unwrap operation (as some unwrap operations create "black" region(s) on the image that should not be taken into account for the stitching).

## EasyBarCode2

- EBarCode supports the PDF417 symbology.

## EasyDeepOCR

- **EasyDeepOCR** exposes the 3 methods that are the core components of `ETextReader.Read` to enable a greater flexibility in designing custom OCR processing workflows:
  - `ETextReader.Localize`
  - `ETextReader.Recognize`
  - `ETextReader.Merge`




## new Open eVision Studio (preview)

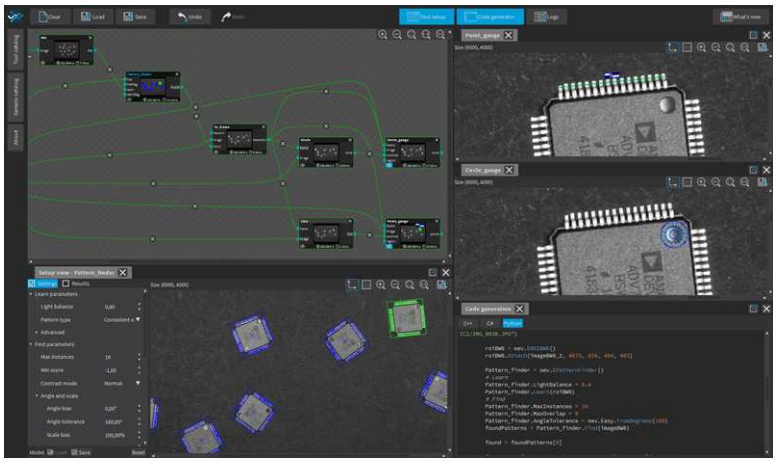
**new Open eVision Studio** includes these new tools:

- Acquisition
  - `3D DepthMap file` to load depth maps.
- **Easy**
  - `Rectangle`, `Circle`, `Polygon` and `Line` to create the respective 2D objects.
  - `To Array` to build an array of `EImage` or ERegion from up to 4 inputs.
  - `Wedge` editor.
  - `Frame` to manually define a frame.
  - `To Frame` to convert objects to a frame.
- **EasyColor**
  - `Color Transform` to apply and create a color lookup table.
- **EasyGauge**
  - `Rectangle gauge`, `Circle gauge`, `Polygon gauge`, `Line gauge`, `Wedge gauge` and `Point gauge` to measure the respective shape.
- **Easy3D**
  - `3D Point`, `3D Box`, `3D Sphere` and `3D Plane` to create the respective 3D objects.
  - `3D Transformer` to apply a 3D transformation on a ZMap, a point cloud or a mesh.
  - `3D Photometric Stereo` to process images using photometric stereo.

New 2D and 3D file converters:

- `3D DepthMap to Point Cloud`
- `3D DepthMap to Mesh Converter`
- `3D DepthMap to image converter`
- `3D ZMap to image converter`
- `3D Image to DepthMap converter`
- `3D Image to ZMap converter`

- **Easy3DLaserLine**
  - `3D Laser Line Scale Calibration` for scale-based laser line calibration.
  - `3D Laser Line Explicit Calibration` for explicit laser line calibration.
  - `3D Laser Line Object Calibration` for object-based laser line calibration.
  - `3D Laser line extractor` to extract a 3D object from a set of laser lines profiles.



# 2. Release Specifications

## OS and processor architectures

### Windows OS

- **Open eVision** is a 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® | 64-bit | |
| Windows 11® IoT Enterprise | 64-bit | on x86_64 systems |
| Windows 10® IoT Enterprise | 64-bit | |

### Linux OS

- **Open eVision** is compatible with x86_64 and aarch64 (ARMv8-A) CPUs.
- **Open eVision** and the **Neo License Manager** are designed to be distribution-independent on x86_64 platforms.
- The minimum requirements are:
  - gcc version 10
  - glibc version 2.27
- This release has been validated with the following distributions and their default gcc compilers and cmake programs:
  - For **deb** based distributions, the following distributions are tested and supported:
    - **Ubuntu 20.04**, **Ubuntu 22.04** and **Ubuntu 24.04**
    - **Debian 11** and **Debian 12**
  - For **rpm** based distributions, the following distributions are tested and supported:
    - **RHEL 9**
    - **Rocky Linux 9**
- This release has been validated on the following embedded systems:
  - **Raspberry** Pi 4, **Raspberry** Pi 5 and **Raspberry** Pi Zero 2 W
  - **NVidia** Jetson Orin series

- For **CMake** samples: the client programs must be linked against the pthread library.
- For **Qt** samples: it is not necessary to make this dependency explicit because a program using Qt automatically depends on the pthread library.

### NVIDIA GPU support

- **Open eVision** uses **CUDA** v12.9 and supports only GPU with a compute capability of 7.5 or more: from architecture Turing (RTX2000 series) to Blackwell (RTX 5000 series).

### 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
  - Virtual machines are supported. **Microsoft Hyper-V**, **Oracle VirtualBox** and **libvirt** hypervisors have been successfully tested.
  - Only the **Neo Licensing System** is compatible with virtualization.

## Supported programming languages and IDEs

The **Open eVision** libraries and tools support C++, Python and the programming languages compatible with the .NET (C#, VB.NET).

### C++ requirements

- A compiler compatible with the C++ 11 standard is required to use **Open eVision**.
- 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.

### .NET requirements

- The .NET framework 4.8 (or later) or the .NET platform 6.0 (or later) is required to use the C# version of **Open eVision**.

### Python requirements

- Python 3.11 or later is required to use the Python bindings for **Open eVision**.
- No other third party dependencies are necessary to use the Python bindings.

### Compatibility with IDEs

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

| | Programming language | |
|---|---|---|
| **IDE** | **C++** | **C#, VB.NET, C++/CLI** |
| 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++ | |

(*) A C++ compiler like **MSVC** must be installed.

- Any **Windows** and **Linux** IDE supporting a compatible compiler (**MSVC 2017** or higher, **GCC 7.5** or higher) should be suitable to build **Open eVision** applications.

# 3. End of Life and Support

## End of life announcements

### OS, programming languages and IDEs

- .NET Framework
  - Starting with **Open eVision 25.02** (February 2025 release) the minimum required version for the .NET Framework will be 4.8.
- Windows OS
  - The support for **Windows 7** and **Windows 8** will stop with **Open eVision 25.10** (October 2025 release). **Open eVision 25.06** will be the latest release supporting **Windows 7** and **Windows 8**.

### Open eVision soft-based / host licensing system

- The **Open eVision** soft-based / host licensing system will be progressively phased out.
  - It was introduced in 2007 with the first version of **Open eVision** and it is based on an old, and now obsolete technology.
  - It is superseded by the **Neo Licensing System**.

**Milestones of the phase out period**

- Since 1 January 2023:
  - There are no more sales of **Open eVision** soft-based / host licenses.
  - The related product codes are 4250 to 4289.
  - These products are removed from the price lists.
- Starting 1 January 2024:
  - The support for the soft-based / host licenses will be removed from the **Open eVision** libraries.
  - The **Open eVision** releases 24.02 and later will not be able to detect and use any soft-based / host licenses activated on the platform.
- Starting 1 January 2029:
  - The soft-based license operation server will be shut down.
  - Activating or recovering a soft-based / host license will not be possible after 2028.

**Notes**

- All applications using a soft-based / host license will continue to work “forever”, as long as the licenses have been activated and don’t require recovery.
- The **Open eVision Neo Licensing System** replaces the soft-based / host licensing system.
- The **Neo Licensing System** supports dongles and **Neo Software Containers** for licenses.
- The usage and features of the **Neo Software Containers** are the same as the old soft-based / host licenses.
- **eVision** licenses are not affected by these changes.

# Support of older environments

## 32-bit libraries

- The **32-bit Open eVision** libraries (only available for **Windows**) were removed from **Open eVision** distributions in July 2023.
  - You should migrate to 64-bit if you need newer versions of **Open eVision**.
  - ► The last version with 32-bit support is **Open eVision 23.04**.

## Soft-based / host licenses

- The support for the soft-based / host licenses is removed from the **Open eVision** libraries.
  - ► The last version to support soft-based / host licenses is **Open eVision 23.12**.

## OS and processor architectures

| OS | | Last version with support |
|---|---|---|
| Windows 8® | 64-bit | **Open eVision** 25.06 |
| Windows 7® (*) | 64-bit | |
| Windows 11® | 32-bit | **Open eVision** 23.04 |
| Windows 10® | 32-bit | |
| Windows 8® | 32-bit | |
| Windows 7® | 32-bit | |
| Windows Vista | — | **Open eVision** 2.3.3.10777 |
| Windows XP | — | **Open eVision** 2.2.2.10255 |
| Windows 2000 | — | **Open eVision** 1.0.1.5222 |

*(*) The recommended version is 6.1.7601 (Windows 7 Service Pack 1)*

## Supported IDE

| IDE | Last version with support |
|---|---|
| Microsoft Visual Studio 2008 | **Open eVision** 22.12 |
| Microsoft Visual Studio 2005 | **Open eVision** 2.7 |
| Microsoft Visual Studio 2003 | |
| Microsoft Visual Studio 6.0 | **Open eVision** 2.5.1.1107 |

## Programming languages

| Programming language | Last version with support |
|---|---|
| Borland C++ Builder 6.0 | **Open eVision** 2.5.1.1107 |
| CodeGear Delphi 2009 | |
| CodeGear C++ Builder 2009 | |
| ActiveX API | |
| Embarcadero RAD Studio XE4 and XE5 | **Open eVision** 2.4.1.11114 |
| Borland Delphi 6.0 and 2006 | **Open eVision** 1.0.1.5222 |
| Borland C++ Builder 2006 | |

## Legacy Headers

- The **Legacy Headers** are removed from **Open eVision** distributions since 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 last version with support is **Open eVision 22.12**.

# 4. Release Details

## 4.1. Features Updates

### New features

#### New reference frame and alignment object EFrame

The new EFrame represents a local reference frame and is used to express various data structures of **Open eVision** such as regions, geometrical primitives, gauges, EFoundPattern, bar codes, matrix codes, QR codes (and more in future releases).

- Use the method ToFrame of a data structure to convert it to a frame.
- You can calibrate the frame so it represents a reference frame in the real world.
- The old class EFrame is now part of the namespace Euresys::Open_eVision::Legacy.

#### EasyGauge and shapes

The **EasyGauge** and the shape API have been reworked.

The differences with the old API are:

- The origin of the image coordinate system of **EasyGauge** is now the top left corner of the top left pixel, that is the same as for the other **Open eVision** tools and libraries.
- The zoom and pan used for the drawing and the graphical manipulation are no longer internal to the gauges. You must give them as parameters of the drawing and graphical manipulation methods.
- The **EasyGauge** API no longer has an internal gauge hierarchy.

  You must now explicitly implement this gauge hierarchy in your code:

  - Convert the measured shape of a gauge to an EFrame.
  - Give this frame to the child gauge.
- EBaseGauge.Measure and GetMeasuredXXX throw an exception if they do not find the shape (in the old API, GetMeasuredXXX returned the nominal shape if the shape was not found).
- The edges of the rectangle and the wedge gauges are defined with, respectively, the enum ERectangleGaugeEdge and EWedgeGaugeEdge.
- The primitive shapes (ERectangle, EPolygon, ECircle, EEllipse, EWedge and ELine) inherit from EShape and implement drawing and graphical manipulation. You can use them as the EXXXShape of the old API.
- The point gauge (EPointGauge) is defined by a nominal line and it no longer has tolerances.
  - The tolerances of the point gauge in the old API are replaced by the length and the angle of the nominal line.
  - The point gauge in the new API also returns a new class EPointSet containing all the measured points.
- EBaseGauge.EditionMode replaces the old "Draggable" and "Rotatable".

- The dragging mode property is removed.
- The optional draw property is replaced by the `enum EDrawingMode_TransitionType` to pass to `EBaseGauge.Draw`.
- To draw the tolerances, call `EBaseGauge.Draw` with `EDrawingMode_Tolerances`.
- The quick draw property is removed.
- The new class `ESamplePath` represents a path with the edge crossings found in the image.
  - The method `EBaseGauge.MeasureSamplePath` returns an object of this new class.
  - Use `ESamplePath` to plot the profile and/or the derivative of the measured path, and get the list of the measured points.
- The new class `ESamplePoint` represents a point on a sample path found by all shape fitting gauges.
- The known issue on using a gauge on an ROI is fixed in the new API.

To continue to use the old API, adapt your code to add the proper legacy namespace (`Euresys::Open_eVision::Legacy`) to each **EasyGauge** object and geometrical primitive.

## Easy3D

- Use `EPhotometricStereoImager` to tune the standard derivation of the Gaussian blur applied before computing the curvatures through the property `GaussianBlurSigma`.
- Use `EPhotometricStereoImager` to query whether the object is calibrated.
- Use `EZMapToMeshConverter` to convert an `EZMap*` to an `EMesh`.
- Use `EDepthMapToMeshConverter` to convert an `EDepthMap*` to an `EMesh`.
- Use `EDepthMapToPointCloudConverter` to convert an `EDepthMap*` to an `EPointCloud`.
- You can now create an `E3DSphere` from 4 points.
- You can set the center of an `E3DBox` after its construction.

## Easy3DLaserLine

- ELaserLineExtractor is now serializable.
- `EObjectBasedCalibrationGenerator` can handle pyramids that are oriented around the negative z-axis using the new property `AlongPositiveZAxis`.

## Easy

- Use the new method `Fit` of `ECircle` to fit a circle (or a circle arc) to a set of points.
- Use the new class `EEllipse` to manipulate ellipses.
  - Use its new method `Fit` to fit an ellipse (or an ellipse arc) to a set of points.
  - The ellipse drawing methods of the draw adapters have variants taking the class `EEllipse` as argument.
  - You can construct the class `EEllipseRegion` from an `EEllipse` and retrieve an `EEllipse` from it.
- Use the new class `ECameraCalibration` for the calibration.
  - It replaces `EWorldShape` with an equivalent API.
  - `EWorldShape` is now in the namespace `Euresys::Open_eVision::Legacy`.

- When unwrapping an image, EWordShape / ECameraCalibration can output an ERegion.
  - Parts of the image that are not contained in the region should not be considered when doing the computation (some unwrap operations create "black" region(s) on the unwrapped image that should not be taken into account).
  - One example of component making use of this new feature is the EImageStitcher that can now take a vector of ERegion as input.

## EasyImage

- EImageStitcher:
  - The new "Incremental Stitching" mode enables the overlapping of the image capture and the stitching operations to reduce the overall processing time while maintaining identical results.
  - In grid mode, the new parameters overlapX and overlapY specify the overlaps between adjacent images on the X and Y axes.
  - Use the new algorithm parameter EImageStitcher.Algorithm to favor speed or robustness.
  - Use the new Learn method to precompute the transformations to apply to the set of images given as input. After a successful Learn, the subsequent calls to the stitcher are faster and use the same transformations.
  - You can now give to the stitcher a vector of ERegion as an additional input. Each region of this vector indicates the portion of the image containing the original image after an Unwrap operation (as some unwrap operations create "black" region(s) on the image that should not be taken into account for the stitching).
- EHDRFusion is compatible with BW8 images.

## EasyBarCode2

- EBarCode supports the PDF417 symbology.


## EasySpotDetector

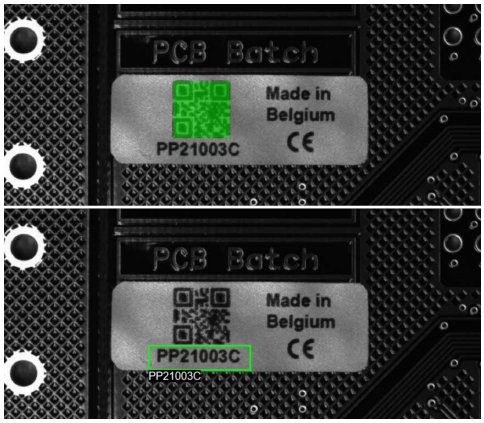
- **EasySpotDetector** has a new C# sample program.

## Deep Learning Studio

- You can now use a previously trained classifier as a pretrained network.
  - This feature is recommended when you want to retrain a network after adding new images to a dataset.

## EasyDeepOCR

- **EasyDeepOCR** exposes the 3 methods that are the core components of `ETextReader.Read` to enable a greater flexibility in designing custom OCR processing workflows:
  - `ETextReader.Localize`
  - `ETextReader.Recognize`
  - `ETextReader.Merge`




## new Open eVision Studio (preview)

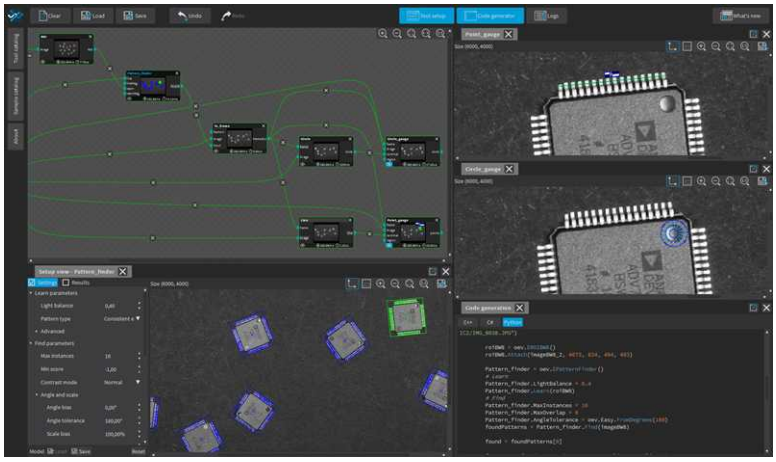
**new Open eVision Studio** includes these new tools:

- Acquisition
  - `3D DepthMap file` to load depth maps.
- **Easy**
  - `Rectangle`, `Circle`, `Polygon` and `Line` to create the respective 2D objects.
  - `To Array` to build an array of `EImage` or ERegion from up to 4 inputs.
  - `Wedge` editor.
  - `Frame` to manually define a frame.
  - `To Frame` to convert objects to a frame.
- **EasyColor**
  - `Color Transform` to apply and create a color lookup table.
- **EasyGauge**
  - `Rectangle gauge`, `Circle gauge`, `Polygon gauge`, `Line gauge`, `Wedge gauge` and `Point gauge` to measure the respective shape.
- **Easy3D**
  - `3D Point`, `3D Box`, `3D Sphere` and `3D Plane` to create the respective 3D objects.
  - `3D Transformer` to apply a 3D transformation on a ZMap, a point cloud or a mesh.
  - `3D Photometric Stereo` to process images using photometric stereo.

New 2D and 3D file converters:

- `3D DepthMap to Point Cloud`
- `3D DepthMap to Mesh Converter`
- `3D DepthMap to image converter`
- `3D ZMap to image converter`
- `3D Image to DepthMap converter`
- `3D Image to ZMap converter`

- **Easy3DLaserLine**
  - `3D Laser Line Scale Calibration` for scale-based laser line calibration.
  - `3D Laser Line Explicit Calibration` for explicit laser line calibration.
  - `3D Laser Line Object Calibration` for object-based laser line calibration.
  - `3D Laser line extractor` to extract a 3D object from a set of laser lines profiles.



**new Open eVision Studio** includes these new sample projects:

- `3D Depth Map Calibration` illustrates loading, calibration and conversion of depth maps.
- `Color system transformation` changes the color system of an input image.
- `Compose` creates a color image from any 3 gray images in any color system.
- `EasyDeepOCR recognition only` recognizes the text inside a rectangle that is built from a QR code detection.
- Several new **EasyGauge** samples.

The interface is improved:

- You can now drag & drop files in the application workspace.

  The allowed file extensions are:

  - oes to load a project.
  - `bmp`, `jpeg`, `jpg`, `jp2`, `j2k`, `j2c`, `png`, `tiff` and `tif` to create an `Image File` tool and load the given image(s).
  - `pcd`, `ply`, `obj`, `xyz` and `csv` to create a `Point Cloud file` tool and load the given point cloud (s).
  - `stl` to create a `Mesh File` tool and load the given mesh(es).

These tools are improved:

- The following tools now support the batch processing:

  - Shape tools
  - Gauge tools
  - ROI
  - To frame
  - To region
  - Region to image
  - Region morphology
  - Region transform
  - Convolution
  - Morphology
  - Image converter
  - Resize
  - Transform
  - Circle warp
  - Inverse circle warp
  - MatrixCode reader
  - QRCode reader
  - BarCode reader

## Improvements

### Linux

- The Readme files describing the setup of the dependencies of **Open eVision** (readme_oev.txt) and the **Neo License Manager** (readme_neo.txt) are now installed in /opt/euresys/Open eVision 25.10/

### Easy

- The **OpenSSL** third-party is updated to version 3.5.1.

### EasyImage

- The class EColorLookup has a copy constructor and an assignment operator.
- EImageStitcher:
  - The speed and the stability of EImageStitcher are greatly improved (it is up to 3 times faster).
  - The memory consumption is greatly improved (it uses up to 3 times less memory).
- SetCircleWarp and SetInvCircleWarp have a new overload that takes an EWedge as argument.
- EasyImage.Copy(image, image) and EasyImage.Copy(constant, image) now benefit from multithreading.
- EasyImage.Copy(EC24, EROIC24) is now faster when the 3 channels have the same value.

### EasyMatch

- The average speed of EMatcher.Match is improved by a factor of 1.2 on Windows and Linux x64 and by a factor of 2.3 on Linux ARM.

### EasyBarCode

- The average speed of EBarCodeReader.Read is improved by a factor of 1.15 on Linux ARM.

### EasyDeepOCR

- Some underlying algorithms that can be used by Optimize are sped up on ARM64.

### Deep Learning tools

- New snippets are available: training an unsupervised and a supervised segmenter.

## new Open eVision Studio (preview)

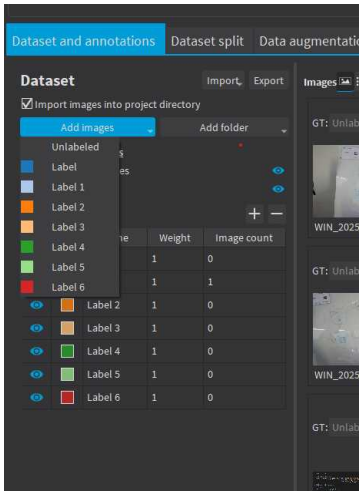
- Interface
  - The dialogs to select files on disk now open to the latest browsed directory.
  - There is a button `Close` in the pop-up windows `What's New`.
  - Additional information is available in 2D display (size for images, size and resolution for ZMaps and depth maps).
  - The tool `Spot detector` has a new button to reset its state.
  - The chronometers measuring the processing time of the tools are now more legible when the tools refresh at a high rate.
  - The propagation of data through the graph is now faster.
- Acquisition
  - `eGrabber studio` tool:
    - The port `output` is renamed `image`.
- **Easy**
  - `Region transform` tool:
    - The tool performs geometrical transformation operations on a region.
  - Unwrap tool:
    - The tool can output an `ERegion` (or an array of `ERegion` depending on the input).
  - `ROI` tool:
    - The port `input` is renamed `image`.
    - Use the new button `Reset` to reset the ROI to a default position.
    - The additional view displays only the inner part of the ROI.
- **EasyImage**
  - `Image stitcher` tool:
    - The additional view properly draws the quadrangles when the option is selected.
    - You can use an array of `region` as input.
    - A new sample showcases the new learning feature.
  - `Circle warp` tool:
    - It includes a full circle checkbox.
    - The parameters `Width` and `Height` are renamed `Number of tangent points` and `Number of radial points` to get closer to the API of **Open eVision**.
    - `Automatic Output Size` is removed as the optimal output values are dependent on your needs and cannot be automatically inferred.
    - The node widget does no longer display the input image and the wedge. It displays the output of the tool, in an effort to standardize with other tools.
    - An additional `wedge` input is available.
    - You can no longer edit the wedge from the setup panel, as it is defined in a new dedicated tool.
    - You cannot load old projects as the wedge information is no longer imported with the tool.
    - The port `input` is renamed `image` and the port `output` is renamed `warp`.

- `Inverse circle warp` tool
  - It includes a full circle checkbox.
  - If `Automatic output size` is checked, the related fields are removed from the setup widget. **Open eVision** infers the optimal output size that is now displayed on the top left corner of the picture.
  - An additional `wedge` input is available.
  - You can no longer edit the wedge from the setup panel, as it is defined in a new dedicated tool.
  - You cannot load old projects as the wedge information is no longer imported with the tool.
  - The port `input` is renamed `image` and the port `output` is renamed `unwarp`.
- `Convolution` tool:
  - You can use a `region` as input.
- `Morphology` tool:
  - You can use a `region` as input.

- **EasyDeepOCR**:
  - `Text reader` tool:
    - You can use the new `Recognize` only `Mode`.
    - You can no longer run an optimization when the sample list is empty.
  - `Get component` tool:
    - You can use a color lookup table as an input.
  - `Compose` tool:
    - You can use a color lookup table as an input.

## Deep Learning Studio

- The OOD results are displayed by label.
- You can filter the images according to their OOD status.
- The button `Add folder` is now available.
- The button `Remove images` is removed. To remove the selected images, right-click on the selection and click on `Remove`.
- In **EasyClassify** and **EasySegment Unsupervised**, the buttons `Add images` and `Add folder` open a drop down menu to select a label.



# 4.2. Breaking Changes

Starting with this release **25.10**, **Open eVision** implements the following changes:

## EasyImage

- `EImageStitcher`:
  - *(25.10.0 - Breaking change)* `GetMatrix` returns a vector of doubles instead of floats.
  - *(25.10.0 - Breaking change)* `MaxFeatures` and `MaxPointMatch` are removed as the new underlying algorithm does not need them anymore.

## EasyGauge and shapes

- *(25.10.0 - Breaking change)* The **EasyGauge** and the shape API have been reworked.
  - To continue to use the old API, adapt your code to add the proper legacy namespace (`Euresys::Open_eVision::Legacy`) to each **EasyGauge** object and geometrical primitive.

For details, see "Features Updates" on page 12.

# 4.3. Changes

Starting with this release **25.10**, **Open eVision** implements the following changes:

## EasyImage

- *(25.10.0)* Use the new parameter `GridOverlap` to set the size of the surface shared by two adjacent patches when the stitcher is in grid mode.
- `EImageStitcher`:
  - *(25.10.0 - Breaking change)* `GetMatrix` returns a vector of doubles instead of floats.
  - *(25.10.0 - Breaking change)* `MaxFeatures` and `MaxPointMatch` are removed as the new underlying algorithm does not need them anymore.

## EasyGauge and shapes

- *(25.10.0 - Breaking change)* The **EasyGauge** and the shape API have been reworked.
  - To continue to use the old API, adapt your code to add the proper legacy namespace (`Euresys::Open_eVision::Legacy`) to each **EasyGauge** object and geometrical primitive.

For details, see "Features Updates" on page 12.

## Regions

- *(25.10.0)* In some cases, the bounding boxes of regions are changed. This can have a minor impact on a processing that uses them.
- *(25.10.0)* The combination of regions (union, subtraction and intersection) keeps a copy of the original region and only performs the rasterization of the combination when needed.
  - Previously, combining two regions rasterized each original region and stored the corresponding combination of the runs.

## Deep Learning redistributables

- *(25.10.0)* The installer for the **Deep Learning** redistributables is updated as **Open eVision** now uses:
  - On the x64 platform: **CUDA** v12.9 and **TensorRT** 10.12
  - On the aarch64 platform: **CUDA** v12.9 and **TensorRT** 10.6 .
- *(25.10.0)* **Open eVision** supports the most recent NVIDIA GPU (Blackwell microarchitecture, RTX5000 series GPU).
- *(25.10.0)* With the "default" and the "TensorRT" engine GPU, **Open eVision** supports from compute capability 7.5 (Turing / RTX2000 series) to 12.1 (Blackwell).

To get the list of GPU for each compute capability or microarchitecture, see https://en.wikipedia.org/wiki/CUDA#GPUs_supported

The engine setup data serialized with previous versions are no longer compatible with this version.

## Deep Learning tools

- *(25.10.0)* The **OpenVINO** engine is upgraded to version 2025.2.0.

## Easy3D

- *(25.10.0)* In `EPointCloud`, `AddPoint`, `AddPoints` and `FillPointsBuffer` ignore points with the values `nan` or `inf`.

## EasyOCV

Following demand, **EasyOCV** is partially re-enabled:

- *(25.10.0)* The following functions are re-enabled in the C# version and enabled for the first time in the Python version for API consistency:

```
- EChecker                  - EDiagnostic
- EOCV                      - ELocationMode
- EOCVChar                  - ELearningMode
- ECharCreationMode         - EDegreesOfFreedom
- EQualityIndicator         - EOCVText
```

- *(25.10.0)* The notable exception is `EOCV.CreateTemplateObjects` that remains accessible only in the C++ version.
- *(25.10.0)* C++ and C# samples are also provided again as well as the demonstration images and models.

The whole **EasyOCV** remains marked as deprecated in all versions.

## Neo License Manager

- *(25.10.0)* The **Neo License Manager** command line and graphical user interface are re-architectured and the command line syntax is rewritten.

  Refer to the **Neo License Manager** documentation (or use `--help`) for more information.

- *(25.10.0)* The **Neo License Manager** graphical user interface is upgraded to Qt 6.

## Sample programs

- *(25.10.0)* All samples now include a copyright notice.

## MSVCS sample programs

- *(25.10.0)* All samples are ported from .Net Framework 4.8 to .Net 6.
- *(25.10.0)* The VS2017 suffix is removed from the samples names.

# Deprecated items

## Easy

- *(25.10.0)* The methods `EPoint.Center` and `EPoint.SetCenterXY` are deprecated as they are redundant with others.

## EasyImage

- *(25.10.0)* The types `EImageC15` and `EImageC16` are now deprecated and will be removed in the next release.

# 4.4. Solved Issues

The following issues have been fixed in **Open eVision 25.10**:

## New Open eVision Studio (preview)

- Interface
  - *(25.10.0)* Each click on a tool in the graph would trigger an undoable action. This has been fixed.
  - *(25.10.0)* Black borders would sometimes appear around the images of the tools in the graph. This has been fixed.
  - *(25.10.0)* On Linux, the application icon was not displayed correctly in the task bar. This has been fixed.
  - *(25.10.0)* On Linux, a crash would happen when saving an image without specifying the desired file extension. This has been fixed.
  - *(25.10.0)* The application would crash when saving a `BW16` image to an unsupported format. This has been fixed.
  - *(25.10.0)* An output port of a tool would not always turn green after selecting it. This has been fixed.
  - *(25.10.0)* The application has a reduced CPU consumption when idle.
  - *(25.10.0)* Floating point numbers would sometimes be represented by an ill-formed scientific notation in the generated code. This has been fixed.
  - *(25.10.0)* The bottom of the graph was sometimes cropped. This has been fixed.
  - *(25.10.1)* A tool performing batch processing crashed when receiving an empty vector as input. This has been fixed.
  - *(25.10.1)* A tool performing batch processing sometimes generated invalid code when its input did not impact the code. This has been fixed.
  - *(25.10.1)* In the `Tool catalog` and the `Sample catalog`, the search field now works properly and the results are displayed in the 'best score' order.
- **Easy**
  - *(25.10.1)* In the shape and gauge tools, drawing a shape with extreme coordinates in a calibrated frame could cause lags and crashes. This has been fixed.
  - *(25.10.1)* In the tool `ROI`, the zoom controls sometimes disappeared when an image is displayed. This has been fixed.
- **EasyImage**
  - *(25.10.0)* In the tool `Image converter`, the Left shift setting was appearing during an `EImageBW16` to `EImageBW16` pass-through. This has been fixed.
  - *(25.10.0)* In the tool `Image converter`, an error would pop when zooming in the image. This has been fixed.
  - *(25.10.0)* In the tool `Arithmetic logic`, the error messages better explains how to fix the issue.

- **EasyObject**
  - *(25.10.0)* When a tool `Object selection` was present in the graph, applying undo/redo in succession caused the application to crash. This has been fixed.
  - *(25.10.0)* With BW16 images, the low threshold of the tool `Image encoder` could not be higher than 255. This has been fixed.
- **EasyGauge**
  - *(25.10.1)* In the gauge tools, the tolerance setting was not included in the generated code. This has been fixed.
  - *(25.10.1)* In the gauge tools, the property `Draw sampled points` was not properly saved. This has been fixed.
  - *(25.10.1)* In the shape and gauge tools, drawing a shape with extreme coordinates in a calibrated frame could cause lags and crashes. This has been fixed.
- **EasyFind**
  - *(25.10.0)* In the tool `Pattern finder`, the property `Interpolate` was not editable. This has been fixed.
- **EasyBarCode**
  - *(25.10.0)* In the tool `BarCode reader`, the value of the keep and add buttons of the learning tab were ignored. This has been fixed.
- **EasyClassify**
  - *(25.10.0)* In the tool `EasyClassify`, the model path setting is homogenized to be more in line with the other tools.
- **EasySpotDetector**
  - *(25.10.0)* In the tool `Spot detector`, whether the alignment was enabled or not was not properly restored on deserialization. This has been fixed.
- Sample programs
  - *(25.10.1)* In the sample programs `EasyDeepOCR reading`, `Region`, `3D ZMap leveler` and `Speed-up processing with resize`, a link was missing between the image file and the region editor tools. This has been fixed.
  - *(25.10.1)* The sample `Classification` now properly loads its images and models.

## Easy

- *(25.10.1)* When a frame was drawn, the tip of the arrow exceeded the axis size. Now the arrow with a tip included is drawn at the size of the axes.
- *(25.10.1)* The method `AutoCalibrateLandmarks` of `ECameraCalibration` and `Legacy.EWorldShape` no longer worked correctly with a boolean argument since 25.06 (the property `CalibrationSucceeded` was always `false`). This has been fixed.

## EasyImage

- *(25.10.0)* The grid mode of `EImageStitcher` works with 1 row or 1 column.
- *(25.10.0)* The `AdaptiveThreshold` with `Median Convolution` filters produced wrong results when the number of threads (set with `Easy.MaxNumberOfProcessingThreads`) was greater than 1. This has been fixed while maintaining the advantage of multithreaded systems.

## Regions

- *(25.10.0)* Calling `ERegion.Prepare` on an empty `EPolygonRegion` threw an exception. This has been fixed.
- *(25.10.0)* Calling `ERegion.Prepare` on a degenerated `EPolygonRegion` could trigger a crash. This has been fixed.
- *(25.10.0)* The bounding box of an `EPolygonRegion` was not updated when removing a point with `EPolygonRegion.RemovePoint`. This has been fixed.

## EasyGauge

- *(25.10.0)* An `EPolygonGauge` could crash when `MeasurementMode` was set to `EPolygonMeasurementMode_Point` and `FilteringThreshold` was too large. This has been fixed.
- *(25.10.1)* The method `MeasureSamplePath` of `ERectangle` returned incorrect results for the top and right edges. This has been fixed.
- *(25.10.1)* Plotting `EPlotItem.Points` after `EPlotItem.Transitions` or `EPlotItem.Peak` did not always work properly. This has been fixed.

## EasyMatch

- *(25.10.0)* Performing an advanced learning of a uniform pattern on a uniform background was very slow. This has been fixed.

## EasyBarCode2

- *(25.10.0)* The decoded string was cut at the first '`\0`' character found, which can happen in some symbologies, for example in `Code128`. This has been fixed.
- *(25.10.1)* In rare occurrences, crashes occurred when reading an ROI with `endX` and `endY` close to the width and height of the parent image. This has been fixed.

## EasyMatrixCode2

- *(25.10.0)* Some false-positives codes were returned with an empty decoded string. This has been fixed and they are not returned anymore.

## EasyOCR2

- *(25.10.0)* On aarch64, the **EasyOCR2** objects had some uninitialized parameters leading to an invalid read (this happened with a very low probability).
  - If an object with uninitialized values was saved in a previous version of **Open eVision**, the **EasyOCR2** object resets them back to their default value.
- *(25.10.0)* The default constructor of the class `EasyOCR2.EOCR2` left some parameters uninitialized, potentially causing issues in very rare situations. This issue has been fixed.
  - Starting with this version, **Open eVision** automatically resets these uninitialized parameters when `EasyOCR2.EOCR2` objects saved with earlier versions are loaded back.

## Deep Learning Studio

- *(25.10.0)* When importing an **Deep Learning** dataset into an existing project, the loaded images could be labeled incorrectly. This has been fixed.
- *(25.10.0)* When importing an **EasyLocate** interest point dataset into an **EasyLocate** interest point project, unnecessary warnings were displayed. This has been fixed.
- *(25.10.0)* When importing an **EasyLocate** interest point dataset into an **EasyLocate** bounding box project, the imported annotations were interest points. This has been fixed.
  - Now, the imported annotation bounds the boxes with a width and height equal to `EClassificationDataset.ObjectSize`.
- *(25.10.0)* Removing an imported image did not delete the corresponding file. This has been fixed.
- *(25.10.0)* In **EasySegment Unsupervised**, you could delete the "good" segmentation label. This has been fixed.
- *(25.10.1)* The labels `Num Inference` and `Throughput` were merged together when exporting a benchmark as a CSV file. This has been fixed.
- *(25.10.1)* The inference precision is now shown as a string (for ex. a `FLOAT32`) instead of a number (for ex. 1) when exporting a benchmark as a CSV file.
- *(25.10.1)* In the tab `Benchmark`, double-clicking a row in the table could crash **Deep Learning Studio**. This has been fixed.
- *(25.10.1)* The inference precision was ignored when performing a benchmark. The precision was always `FLOAT32`. This has been fixed.
- *(25.10.1)* The names of the benchmarks were only checked when renaming an existing benchmark. The names of the new benchmarks are now checked as well and must be unique per tool.
- *(25.10.1)* Changing the color of a label did not update its color in the contextual menus `Add images` and `Add folder`. This has been fixed.
- *(25.10.1)* The version number is added to the window title.

## Deep Learning tools / EasyDeepOCR

- *(25.10.0)* The engine setup data for the engine `EasyDeepLearningEngine_TensorRT` was not compatible between different platforms (for ex. Linux and Windows) leading to crashes when using a model whose engine setup data was created on a different platform. This has been fixed.
  - Starting with this version, the engine setup data created on a different platform is ignored.
- *(25.10.1)* A crash could occur when initializing multiple neural networks with the **TensorRT** engine in parallel. This has been fixed.

## Deep Learning tools

- *(25.10.0)* Running multiple models in parallel with the OpenVINO engine caused performance issues. This has been fixed.
- *(25.10.0)* The classification models other than "normal" sometimes output an OOD score of 1 for all images. This has been fixed.

## EasyDeepOCR

- *(25.10.0)* In the Python version, the import `open_evision.EasyDeepOCR` that was not available has been fixed.

## EasyClassify

- *(25.10.0)* With non-default engine, making inference with batches of varying sizes could cause a crash in the application. This has been fixed.

## Easy3D

- *(25.10.0)* Destroying an object `E3DPhotometricStereoImager` that was constructed through its copy constructor could cause an undefined behavior. This has been fixed.

## C# version

- *(25.10.0)* In any structure in the API, some methods intended to alter the state of the structure had no effect. This has been fixed.

  The following methods are known to have been affected by this issue:
  - `EPoint.Center`
  - `EPoint.SetCenterXY`
  - `EQuadrangle.SetPoint`
  - `EPen.Load`
  - `EBrush.Load`
- *(25.10.1)* The C# VimbaX sample program did not stop correctly when it did not found a camera. This has been fixed.

# 5. Known Issues

## All

- *(25.02.1)* Since version 24.02, **Open eVision** cannot read back the serialization of some objects made with previous versions.
  - `ERectangleRegion` or `E3DObject` are known to be affected by this issue.

## eGrabber and VimbaX

- *(25.10.0)* To use **eGrabber** or **VimbaX** with **Open eVision** in C++, use the full header (`Open_eVision.h`) as there is currently an incompatibility with the per-library headers.

## Open eVision License Manager

- The **Open eVision License Manager** might not start if the **.NET Framework 4.8** is not installed.
- 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.

## Neo License Manager

- *(24.06.0)* In some cases, offline to offline reactivation may not work.

## Deep Learning tools

- *(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.

## Samples

- *(25.02.0)* In the provided Python samples using both **Open eVision** and **Qt**, a rare bug was reported causing failure at loading **Open eVision** shared library due to negative interaction with the Qt library on some computers only.
  - The conditions necessary to reproduce this bug are currently unknown.
  - A workaround and an explanation of this bug is now provided in those Python / Qt samples.

## 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.

  ⚠ 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.

## 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

  ✓ 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.

## C# version

*(24.10.2)* Multiple methods taking arrays as `ref` parameters (for example: `EClassifier.Classify`) may generate random crashes of the application. This is a regression introduced following changes made in release 24.10.0.

## 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.

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.

## EasyBarCode

- EasyBarCode requires that a quiet zone of at least one full module is present around the whole bar code to be read.

## 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.

## EasyImage

- *(24.10.2)* The constructor `EKernel` using an `EKernelType` as parameter does not use the same rectifier as the associated method `Convol`.
  - For instance, `EKernel(EKernelType_GradientX)` returns a kernel with a `EKernelRectifier_KeepPositive` rectifier while the method `ConvolGradientX` uses a `EKernelRectifier_Absolute` rectifier.
- *(24.10.2)* The method `ConvolKernel` does not have the same border value behavior (`constant`) as the pre-defined method like `ConvolGradientX` which is `mirror`.

## EasyObject

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

## EasyMatch

- 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

The following known issues only refer to the Legacy API of **EasyGauge** (in the namespace `Euresys::Open_eVision::Legacy`).

- 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.

- *(25.06.0)* When attaching a gauge to a `worldshape` or another gauge, you must keep a reference of this gauge alive (that is keep the gauge in a variable or an array in your program) otherwise it "disappears" from the hierarchy (immediately in C++ , depending on the garbage collector in Python and C#).

## 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.

## Easy3D

- *(2.16.0)* When using the class `EMeshToZMapConverter` without extension, some triangles of the mesh close to the ZMap border may be removed.
- *(22.04.1)* The Linux virtual machines do not support EDL in the `E3DViewer`.
- *(24.06.0)* When using the Python wrapper, if EDL is enabled on Linux (x64 and arm) platforms, the `E3DViewer` does not work properly.

## 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.

- *(24.10.0)* In the tab `EasyImage`, in the tool `Image Statistics`, the code generated when using a mask is deprecated.
- *(24.10.2)* In the convolution tool, due to the know issues mentioned for **EasyImage**, the behavior between a predefined kernel and a custom kernel having the same parameters can be slightly different.

## 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.

  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.