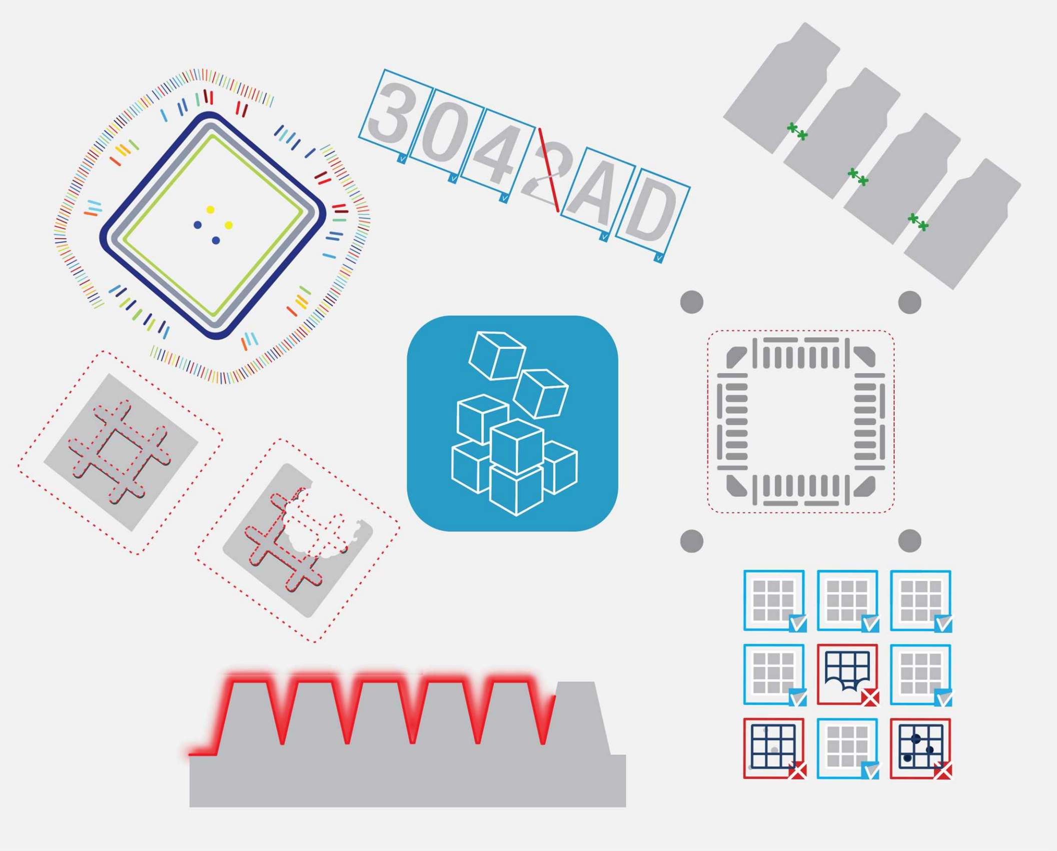
RELEASE NOTES

Open eVision

Release 26.02.0



This documentation is provided with **Open eVision 26.02.0** (doc build **1229**).
www.euresys.com

Contents

1. Release Benefits

What's new in **Open eVision 26.02**

new Open eVision Studio (preview)

New tools:

- Acquisition
  - □ `CVB GenICam Browser` connects the Common Vision Blox GenICam Browser from Stemmer to the **new Open eVision Studio**.
- Easy
  - □ `Image to region` creates an ERegion from an image.
  - □ `Point` creates an EPoint.
  - □ `Frame to point` converts an EFrame to an EPoint.
- EasyImage
  - □ `Uniformize` uses the operation `EasyImage.Uniformize`.
  - □ `Canny edge detector` runs the Canny edge detector on an image.
  - □ `Gain offset` applies gain and offset operations on an image.
  - □ `Threshold` applies a thresholding on an image.
- EasyColor
  - □ `White balance` applies white balance operations on an image.
- Easy3D
  - □ `3D Point Cloud filter` filters a point cloud.
  - □ `3D Filter` filters ZMaps and depth maps.
  - □ `3D Plane transform`, `3D Box transform` and `3D Sphere transform` transform the corresponding form.
  - □ `3D Simple cropper`, `3D Rectangular cropper`, `3D Spherical cropper` and `3D Plane cropper` crop the corresponding form from point clouds.
  - □ `3D Plane fitter` and `3D Sphere fitter` fit a plane or a sphere on a point cloud.
  - □ `3D Principal axis extractor` extracts the principal axes of a point cloud.
  - □ `3D ZMap point extractor` extracts 3D points from a ZMap.
  - □ `3D Features aligner` aligns 2 point clouds or meshes.
- EasyDeepLearning
  - □ `EasyLocate` performs the inference with an **EasyLocate** trained model.
  - □ `EasySegment supervised` performs the inference with an **EasySegment Supervised** trained model.

Tools supporting frames:

- The following tools now have additional frame inputs:

  - □ `EasyFind`
  - □ `EasyMatch`
  - □ `EasyDeepOCR`
  - □ `EasyQRCode`
  - □ `EasySpotDetector`
  - □ `EasyBarCode`
  - □ `EasyMatrixCode`
  - □ `EasyObject`
  - □ Easy:
    - `To region`
    - `Frame` to define a frame within another frame
  - □ EasyImage:
    - `Circle warp`
    - `Inverse circle warp`
    - `Morphology`
    - `Convolution`
    - `Transform`

EasyDeepOCR

- Use the new method `ETextLine.Score` to get a very rough approximation of the reading quality of the underlying text. It returns a score for the returned text line.

Deprecated items

3D Studio

- **3D Studio** has been removed from the **Open eVision** package.
  - ▶ From now on, we recommend prototyping your 3D pipeline in the **new Open eVision Studio**.
  - ▶ **Open eVision** 25.10.1 is the last version that includes the **3D Studio** application.

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 22.04** and **Ubuntu 24.04**
    - **Debian 11** and **Debian 12**
  - For **rpm** based distributions, the following distributions are tested and supported:
    - **RHEL 10**
- 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.
- Since 1 January 2024:
  - □ The support for the soft-based / host licenses is removed from the **Open eVision** libraries.
  - □ The **Open eVision** releases 24.02 and later are 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**.

3D Studio

- **3D Studio** is removed from the **Open eVision** package since the end of 2025.
  - ▶ We recommend prototyping your 3D pipeline in the **new Open eVision Studio**.
  - ▶ The last version that includes the **3D Studio** application is **Open eVision 25.10.1**.

4. Release Details

4.1. Features Updates

New features

Easy

- The method Easy.OnlineDocumentationUrl has been added.
- Use the new methods EFrame.IsDirect and EFrame.IsLocalToSensorDirect to check if the transformation to world and sensor coordinates of the frame is direct.

EasyFind and EasyMatch

- You can now give EFoundPattern.ToFrame the frame in which the pattern was detected as optional argument.
- You can now use Learn / LearnPattern and Find / Match with frame inputs.

EasyMatrixCode

- The method IsFlipped has been added.

Code Readers

- With EMatrixCode, EBarCode, EQRCode and ECode, use the new methods Center, SizeX, SizeY and Angle to retrieve these positional information more easily.
- You can now give EMatrixCode, EBarCode, EQRCode and ECode.ToFrame the frame in which the pattern was detected as optional argument
- You can now give EMatrixCodeReader, EBarCodeReader, EQRCodeReader and ECodeReader.Read an optional EFrame parameter.

EasyOCR2

- **EasyOCR2** now supports EFrame.

Easy3D

- With ZMaps, use the new method ZMapPositionFrame to retrieve a Frame containing the transform from image to ZMap coordinates.
- With ZMaps, use the new method ProjectionPlane to retrieve the projection plane used for the ZMap.
- Use the new method OrientNormal of E3DPlane to orient the plane alongside each of the axes.
- With any ZMap class, use the new overload of ImageToWorld to convert a 2D point to a 3D world point.
- EFeatureAligner now supports unordered points through the parameter AlignmentType. With unordered points, use the parameters X / Y / ZAngleRange to add constraints on the transform.

Deep Learning tools

- New early stopping feature: When training a neural network, if you do not specify an iteration parameter to the Train method of a deep learning tool, the tool now automatically stops the training once it is done.

Deep Learning Studio

- By default, the early stopping feature is activated for all new created tools.
  - The number of iterations parameter is moved to the panel Advanced parameters.
  - To disable early stopping and specify the number of iterations, uncheck the new checkbox Automatic.

EasyLocate

- With ELocatorResult.Draw or ELocatorObject.Draw, you can now use the optional argument highlightObject to control whether the method Draw highlights the object or not. By default, the highlighting is disabled.
- Use the additional methods ELocatorObject.Contains and ELocatorResult.FindObject to identify an object containing a given location.

EasyDeepOCR

- Use the new method ETextLine.Score to get a very rough approximation of the reading quality of the underlying text. It returns a score for the returned text line.

Samples

- With the MSVC sample EasyFindVectorLearn, you can now save the learned pattern finder and zoom in the image.

new Open eVision Studio (preview)

New tools:

- Acquisition
  - CVB GenICam Browser connects the Common Vision Blox GenICam Browser from Stemmer to the **new Open eVision Studio**.

- Easy
  - □ `Image to region` creates an ERegion from an image.
  - □ `Point` creates an EPoint.
  - □ `Frame to point` converts an EFrame to an EPoint.
- EasyImage
  - □ `Uniformize` uses the operation `EasyImage.Uniformize`.
  - □ `Canny edge detector` runs the Canny edge detector on an image.
  - □ `Gain offset` applies gain and offset operations on an image.
  - □ `Threshold` applies a thresholding on an image.
- EasyColor
  - □ `White balance` applies white balance operations on an image.
- Easy3D
  - □ `3D Point Cloud filter` filters a point cloud.
  - □ `3D Filter` filters ZMaps and depth maps.
  - □ `3D Plane transform`, `3D Box transform` and `3D Sphere transform` transform the corresponding form.
  - □ `3D Simple cropper`, `3D Rectangular cropper`, `3D Spherical cropper` and `3D Plane cropper` crop the corresponding form from point clouds.
  - □ `3D Plane fitter` and `3D Sphere fitter` fit a plane or a sphere on a point cloud.
  - □ `3D Principal axis extractor` extracts the principal axes of a point cloud.
  - □ `3D ZMap point extractor` extracts 3D points from a ZMap.
  - □ `3D Features aligner` aligns 2 point clouds or meshes.
- EasyDeepLearning
  - □ `EasyLocate` performs the inference with an **EasyLocate** trained model.
  - □ `EasySegment supervised` performs the inference with an **EasySegment Supervised** trained model.

<u>Tools supporting frames:</u>

- The following tools now have additional frame inputs:
  - □ `EasyFind`
  - □ `EasyMatch`
  - □ `EasyDeepOCR`
  - □ `EasyQRCode`
  - □ `EasySpotDetector`
  - □ `EasyBarCode`
  - □ `EasyMatrixCode`
  - □ `EasyObject`
  - □ Easy:
    - `To region`
    - `Frame` to define a frame within another frame
  - □ EasyImage:
    - `Circle warp`
    - `Inverse circle warp`
    - `Morphology`
    - `Convolution`
    - `Transform`

New sample projects:

- `White balance / gamma correction` applies a gamma correction in order to adjust the brightness and contrast of a dark photograph.
- `Shading correction` increases contrast and improves the visibility of defects or anomalies.
- `EasyLocate (bounding boxes)` identifies and locates electronic components in a small bag through axis-aligned bounding boxes.
- `EasyLocate (interest points)` identifies and locates capacitors using interest points.
- `Supervised segmentation` identifies defects in coffee beans and segments them.
- `Reading circular text` detects a fiducial on a bottle cap using **EasyFind** and uses the frame of reference associated with the result to unwarp and read the circular text on the bottle cap with **EasyDeepOCR**.
- `Learn and find in a frame` uses **EasyFind** to check if a lottery ticket is a winner. On each ticket image, after an initial alignment, a pattern is learned on a region and then searched on another region.
- `3D sphere fitter` removes the background plane, decimates and fits a sphere.
- `3D LLE with text reading` loads profiles, extracts the depth maps and reads the embossed text on the depth maps.
- `3D component check` finds the PCI port on a card with **EasyMatch** and inspect the relative position of the expected components with rectangle gauges.
- `3D remote control inspection` finds remotes with a rectangle gauge, uses **EasyObject** to count the buttons and **EasyGauge** to check the size of one of them.
- `3D calibration and measure` loads and calibrates a depth map, finds the background plane, projects it on a ZMap and measures a circle in the metric space.
- `3D alignment from 2D alignment` finds 2D feature points in a ZMap and performs a 3D alignment by using the corresponding 3D feature points.
- `3D principal axis extractor` removes the background plane to isolate an object and computes its principal axis to align it on the origin of the referential.

These tools are improved:

- Acquisition
  - □ `eGrabber studio` can output `EDepthMap8` and `EDepthMap16`.
- **Easy**
  - □ `To array` supports `EFrame` and `EPoint` as input.
  - □ `To frame` accepts a "frameIn" input when the input `input` is an `EFoundPattern`, an `EBarCode`, an `EMatrixCode` or an `ECode`.
- **Easy3D**
  - □ `3D ZMap to image converter` outputs the frame and the `ERegion` corresponding to the ZMap.
  - □ In `3D Sphere`, you can pick the center of the sphere.
  - □ In `3D Sphere` and `3D Plane`, use the button `Pick all points` to pick the 3 or 4 points in a row.
  - □ `Photometric Stereo` supports vectors of ROI as input and several outputs can be connected at the same time.
  - □ `3D transform` allows operations on input transformation, like inversion and combination.
  - □ `3D Plane finder` exposes the inliers and outliers clouds as outputs.
  - □ `3D Plane finder` has settings to show or hide each components in the 3D display widgets.
- **EasyBarCode**, **EasyMatrixCode** and **EasyQRCode**
  - □ `BarCode reader`, `MatrixCode reader` and `QRCode reader` display the number of read codes.

The interface is improved:

- □ A splash screen is now displayed when the program starts.
- □ The program icon has been refined.
- □ New log messages are added about an incorrect installation.
- □ In the `Tool catalog`, using the search field sets the category filter to `All tools` and selecting a category clears the search field.
- □ in the setup panels, the 2D and 3D option icons in 2D are unified.
- □ The tool nodes can now accept up to 6 inputs or outputs.
- □ In the tool node containers, input and output ports are colored to indicate their status:
  - Green = mandatory
  - Orange = optional
  - Red = disabled
- □ The chronometers now display only 3 significant digits for a better readability and more stability in live mode.

Improvements

All

- On Windows, all **Open eVision** binaries and installers now have a digital signature.

Easy

- For the calibration, you can use the new error values ECameraCalibration.MeanSensorToWorldError, ECameraCalibration.MaxSensorToWorldError, ECameraCalibration.MeanWorldToSensorError and ECameraCalibration.MaxWorldToSensorError.
  - □ The "sensor to world" errors are similar to ECameraCalibration.GridPointsMeanVariation but they are not normalized with respect to grid pitches.
  - □ The score ECameraCalibration.ReprojectionScore is the square of ECameraCalibration.MeanWorldToSensorError.
  - □ The "mean" errors are root mean square errors.
- When used with frames, the precision of ERegion.Transform, ERegion.Invert and ERectangleRegion are improved.
- Use the new method ERegion.SensorToLocal to map a region defined with respect to the image into a local frame of reference.
- The support for negative scales in frames is improved. Use a negative scale to inverse the direction of one axis.
- Use the new overloads to EFrame.Draw, EFrame.Drag and EFrame.HitTest to select the parent frame in which the frame to draw, drag or hit is located.

EasyImage

- These methods now supports frames:
  - □ EasyImage.SetCircleWarp and EasyImage.SetInvCircleWarp.
  - □ The convolution operators EasyImage.Convol×××.
  - □ The morphological operators EasyImage.Dilate×××, EasyImage.Erode×××, EasyImage.Open×××, EasyImage.Close×××, EasyImage.WhiteTopHat×××, EasyImage.BlackTopHat××× and EasyImage.MorphoGradient×××.
  - □ The thresholding operators EasyImage.Threshold, EasyImage.DoubleThreshold, EasyImage.AutoThreshold and EasyImage.AdaptiveThreshold.
  - □ EasyImage.Copy, EasyImage.Median, EasyImage.Histogram, EasyImage.PixelCompare, EasyImage.PixelMax and EasyImage.PixelMin.
  - □ EasyImage.ScaleRotate (the source image pivot, scales, rotation and, optionally, the region are interpreted using the frame).

EasyObject

- **EasyObject** now supports frames. All the floating point features are now expressed in the frame of reference used to encode the image.

EasyFind and EasyMatch

- When slightly moving an ROI, one or more patterns were sometimes no longer detected or were detected at a slightly different position.
  - ▶ The detection is now insensitive to an ROI move.
  - ▶ On average, this change has a negligible impact on the speed.

EasyMatrixCode2

- When slightly moving an ROI, one or more matrix codes were sometimes no longer detected or were detected at a slightly different position.
  - ▶ The detection is now insensitive to an ROI move.
  - ▶ On average, the reader is now 5% slower when using an ROI.

EasyBarCode2

- When slightly moving an ROI, one or more bar codes were sometimes no longer detected or were detected at a slightly different position.
  - ▶ The detection is now insensitive to an ROI move.
  - ▶ On average, the reader is now 8% slower when using an ROI.

Easy3D

- `E3DFilters` now support `ERegions` for noise removal operations and `EFrames` for both noise removal and median operations.
- `E3DBox.Transform` now supports transform matrices with scaling.
- `E3DSphere.Transform` now supports transform matrices with isotropic scaling.

Deep learning tools

- To change or retrieve the color and the opacity of labels, you can now use the API and the methods `EClassificationDataset.Set×××LabelColor`, `EClassificationDataset.Get×××LabelColor`, `EClassificationDataset.Set×××LabelOpacity` and `EClassificationDataset.Get×××LabelOpacity`.

EasyClassify

- The method `Classify` now supports frames.

EasySpotDetector

- When slightly moving an ROI, one or more scratches or spots were sometimes no longer detected or were detected at a slightly different position.
  - ▶ The detection is now insensitive to an ROI move.
  - ▶ This improvement may require changing some parameters when the searched spots or scratches are very small.
  - ▶ On average, this change has a negligible impact on the speed.

new Open eVision Studio (preview)

- All
  - □ In the generated code, if a license is already specified, the required licenses comment no longer displays `Any`".
  - □ You can now use both comma and point characters as decimal separator in numbers.
- **Easy**
  - □ The tool `To frame` now supports an `ECircle` as input.
  - □ The result pane of the tool `Dot grid calibration` now displays the mean and max errors `Sensor to world` and `World to sensor`.
  - □ In the tool `Region editor` and the shape tools, a message is now displayed when the element drawn is too large as when connecting a frame input with large scales.
- **EasyImage**
  - □ The tool `Transform` now takes the region into consideration when fitting the size of the transformed image to the transformation (the transformed image size is now large enough to show the transformed region).
- **EasyFind** and **EasyMatch**
  - □ The error logs are improved.
  - □ You can now sort the results on every column.
- Sample programs
  - □ The sample programs `Speed-up processing with resize` and `Batch processing find and gauge` are now in auto-play mode.

4.2. Breaking Changes

Starting with this release **26.02**, **Open eVision** implements the following changes:

All - ROI and methods Draw

(26.02.0 - Breaking change)

- When the processed image is an ROI, the drawing methods now draw in the referential of the ROI instead of in the referential of the parent image.
  - □ For example, if an object is detected at offset (0, 0) in an ROI that was attached to an image at offset (100, 200), the object is now drawn at offset (0, 0) instead of at offset (100, 200) previously.
- The following methods are updated:
  - □ EQRCode.Draw, DrawErrors, DrawErrorsWithCurrentPen, DrawGrid, DrawPosition and DrawWithCurrentPen
  - □ EasyBarCode2.EBarCode.DrawPosition and DrawPositionWithCurrentPen
  - □ EMatrixCode.Draw, DrawErrors, DrawErrorsWithCurrentPen and DrawWithCurrentPen
  - □ EasyMatrixCode2.EMatrixCode.DrawErrors, DrawErrorsWithCurrentPen, DrawGrid, DrawGridWithCurrentPen, DrawPosition and DrawPositionWithCurrentPen
  - □ ECode.DrawPosition and DrawPositionWithCurrentPen
  - □ EFoundPattern.Draw and DrawWithCurrentPen
  - □ EMatcher.DrawPosition, DrawPositions, DrawPositionsWithCurrentPen and DrawPositionWithCurrentPen
  - □ EOCR.DrawChar, DrawChars, DrawCharsWithCurrentPen, DrawCharWithCurrentPen, DrawObjects, HitChar and HitChars
  - □ EOCR2.DrawDetection, DrawDetectionWithCurrentPen, DrawRecognition, DrawRecognitionWithCurrentPen, DrawSegmentation and DrawSegmentationWithCurrentPen
  - □ EHarrisInterestPoints.Draw, DrawCorner, DrawCornerWithCurrentPen and DrawWithCurrentPen
  - □ EasyOCV.DrawText, DrawTexts, DrawTextChars, DrawTextsChars, DrawTextWithCurrentPen, DrawTextsWithCurrentPen, DrawTextCharsWithCurrentPen and DrawTextsCharsWithCurrentPen
  - □ ECodedImage.DrawObject, DrawObjectFeature, DrawObjects, DrawObjectsFeature, DrawObjectWithCurrentPen, DrawObjectFeatureWithCurrentPen, DrawObjectsWithCurrentPen and DrawObjectsFeatureWithCurrentPen
  - □ ECodedImage2.Draw, DrawFeature, DrawHole, DrawHoleFeature, DrawObject, DrawObjectFeature, DrawWithCurrentPen, DrawFeatureWithCurrentPen, DrawHoleWithCurrentPen, DrawHoleFeatureWithCurrentPen, DrawObjectWithCurrentPen and DrawObjectFeatureWithCurrentPen

- To obtain the previous behavior, change your code as follows:

```
EImageBW8 image;
EROIBW8 roi;
roi.attach(&image, orgX, orgY, width, height);

EObject object;
object.Process(roi);

EDrawAdapter drawAdapter(&image);
object.Draw(drawAdapter, 1, 1, orgX, orgY); // equivalent to previous behavior of object.Draw(drawAdapter, 1,
1, 0, 0);
```

Easy

- *(26.02.0 - Breaking change)* EFourierImage.LoadImage is renamed EFourierImage.Load.
- *(26.02.0 - Breaking change)* Easy.CheckLicense can only check for **Open eVision** licenses and not for **eGrabber** licenses as previously.
- *(26.02.0 - Breaking change)* The key parameters of Easy.SetOemKey and Easy.CheckOemKey are changed to a type appropriate for use in all languages:
    - C++: std::vector<uint8_t>
    - C#: byte[]
    - Python: list[byte]

Regions:

- *(26.02.0 - Breaking change)* Rotate, Scale and Translate are now methods of the base class ERegion and return a std::unique_ptr<ERegion> (in C++) or an ERegion (in C# and Python).
    - Previously, these methods only existed for rectangle, polygon, circle and ellipse regions and usually returned the same type.
- *(26.02.0 - Breaking change)* In ERegion.Transform, ERegion.Rotate and ERegion.Scale, the rotation and scaling center is the gravity center of the region for a run-based region and the gravity center of the vertices for a polygon region.
    - Previously, the rotation center was the center of the bounding box.
- *(26.02.0 - Breaking change)* EFrame.IsDirect is renamed EFrame.IsCalibrationDirect.
    - Note that EFrame.IsDirect still exists but now checks the local to world transformation instead of the calibration.

EasyImage

- *(26.02.0 - Breaking change)* The threshold method for EBW16 is changed so that its parameters are the same as for EBW8.

EasyBarCode2

- *(26.02.0 - Breaking change)* The return type of EBarCodeGradingParameters.ConvertToAlphabeticGrade is changed to a type appropriate for use in all languages:
    - C++: std::string
    - C#: string
    - Python: str

Easy3D

- *(26.02.0 - Breaking change for the Python wrapper)* In the methods `ERectangularCropper.Crop` and `ESphericalCropper.Crop`, the argument `invertCrop` is renamed `keepInside`.

Deep Learning tools

- *(26.02.0 - Breaking change)* The OOD score distribution could be heavily skewed towards 0 or 1, making it impossible to find an OOD threshold. This has been fixed.
  - This update breaks the compatibility with previous OOD models. You can load them into **Open eVision** 26.02 but they are incompatible with previous versions.

EasySpotDetector

- *(26.02.0 - Breaking change)* When slightly moving an ROI, one or more scratches or spots were sometimes no longer detected or were detected at a slightly different position.
  - The detection is now insensitive to an ROI move.
  - This improvement may require changing some parameters when the searched spots or scratches are very small.
  - On average, this change has a negligible impact on the speed.

4.3. Changes

Starting with this release **26.02**, **Open eVision** implements the following changes:

All - ROI and methods `Draw`

(26.02.0 - Breaking change)

- When the processed image is an ROI, the drawing methods now draw in the referential of the ROI instead of in the referential of the parent image.
  - □ For example, if an object is detected at offset (0, 0) in an ROI that was attached to an image at offset (100, 200), the object is now drawn at offset (0, 0) instead of at offset (100, 200) previously.
- The following methods are updated:
  - □ `EQRCode.Draw`, `DrawErrors`, `DrawErrorsWithCurrentPen`, `DrawGrid`, `DrawPosition` and `DrawWithCurrentPen`
  - □ `EasyBarCode2.EBarCode.DrawPosition` and `DrawPositionWithCurrentPen`
  - □ `EMatrixCode.Draw`, `DrawErrors`, `DrawErrorsWithCurrentPen` and `DrawWithCurrentPen`
  - □ `EasyMatrixCode2.EMatrixCode.DrawErrors`, `DrawErrorsWithCurrentPen`, `DrawGrid`, `DrawGridWithCurrentPen`, `DrawPosition` and `DrawPositionWithCurrentPen`
  - □ `ECode.DrawPosition` and `DrawPositionWithCurrentPen`
  - □ `EFoundPattern.Draw` and `DrawWithCurrentPen`
  - □ `EMatcher.DrawPosition`, `DrawPositions`, `DrawPositionsWithCurrentPen` and `DrawPositionWithCurrentPen`
  - □ `EOCR.DrawChar`, `DrawChars`, `DrawCharsWithCurrentPen`, `DrawCharWithCurrentPen`, `DrawObjects`, `HitChar` and `HitChars`
  - □ `EOCR2.DrawDetection`, `DrawDetectionWithCurrentPen`, `DrawRecognition`, `DrawRecognitionWithCurrentPen`, `DrawSegmentation` and `DrawSegmentationWithCurrentPen`
  - □ `EHarrisInterestPoints.Draw`, `DrawCorner`, `DrawCornerWithCurrentPen` and `DrawWithCurrentPen`
  - □ `EasyOCV.DrawText`, `DrawTexts`, `DrawTextChars`, `DrawTextsChars`, `DrawTextWithCurrentPen`, `DrawTextsWithCurrentPen`, `DrawTextCharsWithCurrentPen` and `DrawTextsCharsWithCurrentPen`
  - □ `ECodedImage.DrawObject`, `DrawObjectFeature`, `DrawObjects`, `DrawObjectsFeature`, `DrawObjectWithCurrentPen`, `DrawObjectFeatureWithCurrentPen`, `DrawObjectsWithCurrentPen` and `DrawObjectsFeatureWithCurrentPen`
  - □ `ECodedImage2.Draw`, `DrawFeature`, `DrawHole`, `DrawHoleFeature`, `DrawObject`, `DrawObjectFeature`, `DrawWithCurrentPen`, `DrawFeatureWithCurrentPen`, `DrawHoleWithCurrentPen`, `DrawHoleFeatureWithCurrentPen`, `DrawObjectWithCurrentPen` and `DrawObjectFeatureWithCurrentPen`

- To obtain the previous behavior, change your code as follows:

```
EImageBW8 image;
EROIBW8 roi;
roi.attach(&image, orgX, orgY, width, height);

EObject object;
object.Process(roi);

EDrawAdapter drawAdapter(&image);
object.Draw(drawAdapter, 1, 1, orgX, orgY); // equivalent to previous behavior of object.Draw(drawAdapter, 1,
1, 0, 0);
```

All

- *(26.02.0)* All **Open eVision** classes have the following pair of methods `Load` and `Save` (some were previously missing):
  - `Load(string)` and `Load(ESerializer)`
  - `Save(string)` and `Save(ESerializer)`

  This means the following classes have new methods:
  - `EImageBW32f`
  - `VectorTypes`
  - `ECalibrationModel`
  - `EFourierImage`
  - `EOCR2CharacterDatabase`

Easy

- *(26.02.0 - Breaking change)* `EFourierImage.LoadImage` is renamed `EFourierImage.Load`.
- *(26.02.0 - Breaking change)* `Easy.CheckLicense` can only check for **Open eVision** licenses and not for **eGrabber** licenses as previously.
- *(26.02.0 - Breaking change)* The key parameters of `Easy.SetOemKey` and `Easy.CheckOemKey` are changed to a type appropriate for use in all languages:
  - C++: `std::vector<uint8_t>`
  - C#: `byte[]`
  - Python: `list[byte]`

Regions:

- *(26.02.0 - Breaking change)* `Rotate`, `Scale` and `Translate` are now methods of the base class `ERegion` and return a `std::unique_ptr<ERegion>` (in C++) or an `ERegion` (in C# and Python).
  - Previously, these methods only existed for rectangle, polygon, circle and ellipse regions and usually returned the same type.
- *(26.02.0 - Breaking change)* In `ERegion.Transform`, `ERegion.Rotate` and `ERegion.Scale`, the rotation and scaling center is the gravity center of the region for a run-based region and the gravity center of the vertices for a polygon region.
  - Previously, the rotation center was the center of the bounding box.
- *(26.02.0 - Breaking change)* `EFrame.IsDirect` is renamed `EFrame.IsCalibrationDirect`.
  - Note that `EFrame.IsDirect` still exists but now checks the local to world transformation instead of the calibration.

EasyImage

- *(26.02.0 - Breaking change)* The threshold method for `EBW16` is changed so that its parameters are the same as for `EBW8`.

EasyFind and EasyMatch

- *(26.02.0)* When slightly moving an ROI, one or more patterns were sometimes no longer detected or were detected at a slightly different position.
  - ▶ The detection is now insensitive to an ROI move.
  - ▶ On average, this change has a negligible impact on the speed.

EasyMatrixCode2

- *(26.02.0)* When slightly moving an ROI, one or more matrix codes were sometimes no longer detected or were detected at a slightly different position.
  - ▶ The detection is now insensitive to an ROI move.
  - ▶ On average, the reader is now 5% slower when using an ROI.

EasyBarCode2

- *(26.02.0)* When slightly moving an ROI, one or more bar codes were sometimes no longer detected or were detected at a slightly different position.
  - ▶ The detection is now insensitive to an ROI move.
  - ▶ On average, the reader is now 8% slower when using an ROI.
- *(26.02.0 - Breaking change)* The return type of `EBarCodeGradingParameters.ConvertToAlphabeticGrade` is changed to a type appropriate for use in all languages:
  - □ C++: `std::string`
  - □ C#: `string`
  - □ Python: `str`

Easy3D

- *(26.02.0 - Breaking change for the Python wrapper)* In the methods `ERectangularCropper.Crop` and `ESphericalCropper.Crop`, the argument `invertCrop` is renamed `keepInside`.
- *(26.02.0)* `EFilters.RemoveNoise` and `EFilters.Median` only threw exceptions with internal error codes. The error messages are now more specific.

Deep Learning tools

- *(26.02.0 - Breaking change)* The OOD score distribution could be heavily skewed towards 0 or 1, making it impossible to find an OOD threshold. This has been fixed.
  - ▶ This update breaks the compatibility with previous OOD models. You can load them into **Open eVision** 26.02 but they are incompatible with previous versions.

EasySpotDetector

- *(26.02.0 - Breaking change)* When slightly moving an ROI, one or more scratches or spots were sometimes no longer detected or were detected at a slightly different position.
  - ▶ The detection is now insensitive to an ROI move.
  - ▶ This improvement may require changing some parameters when the searched spots or scratches are very small.
  - ▶ On average, this change has a negligible impact on the speed.

EasyDeepOCR

- *(26.02.0)* Using the method `Recognize` of the reader with a region sometimes returned very different geometric results when the region was slightly modified. To ensure the oriented rectangle obtained from the region is deterministic, **EasyDeepOCR** now throws an exception if neither the angle hint nor the automatic reading direction is set.

License manager

- *(26.02.0)* The license managers version numbers are now independent of the **Open eVision** version number.
  - □ The version 3.1 is distributed along **Open eVision** 26.02.
- *(26.02.0)* The license managers are now installed in a folder without version reference.

Linux installers

- *(26.02.0)* The installers now include the following dependencies:
  - □ **Open eVision** `x86_64 .deb` installer: `libfreetype6` and `libharfbuzz-bin`
  - □ **Open eVision** `arm64 .deb` installer: `libfreetype6`, `libharfbuzz-bin` and `libgomp1`
  - □ **Open eVision** `x86_64 .rpm` installer: `freetype` and `libxkbcommon`
- *(26.02.0)* The Linux dependencies installation is simplified by specifying optional third-party dependencies in the installation packages.
  - □ Those packages are installed by default when using the usual commands `apt-get install` or `dnf install`.
  - □ To perform an installation with minimal third-party dependencies, follow the instructions in the Linux `readme` file.

new Open eVision Studio (preview)

- *(26.02.0)* The shape tools `3D ×××` do not take a `3DTransformMatrix` as input anymore as this is now handled by the new tools `3D ××× transform`.
- *(26.02.0)* The tools `3D ××× transform` now apply the rotation before the scaling.
- *(26.02.0)* The type of an image, a ZMap or a DMap is now displayed in the status bar of the 2D display widget.
- *(26.02.0)* The results of a `Pattern Finder` and of a `Matcher` and the grades of a `QRCode reader` and of a `MatrixCode reader` are not displayed in scientific notation anymore.
- *(26.02.0)* The port names of the following tools are renamed (without breaking backward compatibility of serialization):
  - □ **Easy**: `Unwarp`, `Frame`, `To frame`, `Region transform`, `Region morphology`, `Dot grid calibration`
  - □ **EasyImage**: `Resize`, `Convolution`, `Morphology`, `Transform`
  - □ **EasyColor**: `Color transform`
  - □ **EasyObject**: `ObjectSelection`,
  - □ **Easy3D**: `3D ZMap leveler`, `3D Point Cloud decimator`, `3D Point Cloud filter`, `3D Transform`
- *(26.02.0)* The **EasyDeepOCR** reader tool can now take an `ERectangle` as its input with the port `region`.

Deprecated items

Easy

- *(26.02.0)* EBaseROI.SetImagePtr is now deprecated.
  - ▶ Instead, use EBaseROI.UseExternalBuffer.
- *(26.02.0)* EFourierImage.SetImagePtr is now deprecated.
  - ▶ Instead, use EFourierImage.UseExternalBuffer.
- *(26.02.0)* In the enum Euresys.Open_eVision.LicenseFeatures.Features, the unused values Gigelink, Recorder and GigelinkRDMA are removed.
- *(26.02.0)* The constructor of EPoint that takes an EFrame as argument is now deprecated.

Easy3D

- *(26.02.0)* EZMap.SetBufferPtr is now deprecated.
  - ▶ Instead, use EZMap.UseExternalBuffer.
- *(26.02.0)* EDepthMap.SetBufferPtr is now deprecated.
  - ▶ Instead, use EDepthMap.UseExternalBuffer.

EasyOCR

- The methods EOCR.CharGetTotalOrgX, CharGetTotalOrgY, CharGetTotalDstX and CharGetTotalDstY are removed.
  - ▶ Instead, you can add the origin of the ROI to the variants without Total (for ex: CharGetOrgX).

EasyImage

- The current values for the enum EFlipAxis are now deprecated.
  - ▶ Instead, use their new counterpart value (such as EFlipAxis_Horizontal) that respect the current naming convention.

4.4. Solved Issues

The following issues have been fixed in **Open eVision 26.02**:

Easy

- *(26.02.0)* Reading a model file for ECameraCalibration or the tool Dot grid calibration in the **new Open eVision Studio** did not apply the correct transformations WorldToSensor and SensorToWorld and also affected the unwarping. This has been fixed.
- *(26.02.0)* The hit box of the rotation handle of an EFrame was slightly shifted to the right compared to its display. This has been fixed.
- *(26.02.0)* Using ERegion.LocalToSensor resulted in a translation error. This has been fixed.
- *(26.02.0)* It was possible to assign any kind of region to a geometrical region (ERectangleRegion...), leading to an incoherent state of the object. Now, when trying to assign a region to a geometrical region, an exception is thrown if the region is not of the same type.
- *(26.02.0)* ERegion.DrawContour did not work properly when a frame was passed as parameter. This has been fixed.
- *(26.02.0)* Defining a frame from another one could result in a frame exhibiting shear (angles are not preserved) that is not supported. Now, when creating such a frame, an exception with the error EError_FrameShear is thrown if the resulting frame would contain shear.

EasyImage

- *(26.02.0)* The supported size for the destination image was inconsistent in EasyImage.ScaleRotate. In the BW8 with no region version, the minimum size for the destination image was 1×1 and, in the other versions, it was 2×2. Now, all versions have a minimum destination image size of 1×1.

EasyFind

- *(26.02.0)* When learning from and finding patterns on synthetic images with interpolation enabled, the computed scores could be incorrect in some cases. This has been fixed.
- *(26.02.0)* When the corresponding learned pattern was an ROI, the learned region of the EFoundPattern was drawn incorrectly. This has been fixed.

EasyBarCode2

- *(26.02.0)* A rarely occurring bug that could prevent the reading of some bar codes has been fixed.

Deep Learning tools

- *(26.02.0)* Initializing models with **TensorRT** and saving them could lead to a crash when loading the model on a computer with the same OS but a different GPU setup. This has been fixed.
- *(26.02.0)* A device with an old version of the NVIDIA driver could sometimes make the application crash. This has been fixed.
  - If the NVIDIA driver is too old for the current version of **Open eVision**, **Deep Learning Studio** displays a message at start-up.
- *(26.02.0)* Conflicting operations could occur while interacting with image overlays while pressing the CTRL key. This has been fixed.

Deep Learning Studio

- *(26.02.0)* Manually interrupting an ongoing training job and computing missing values set the OOD threshold to 0. This has been fixed.
- *(26.02.0)* Removing all tools would not clear the tab `Validation and results`, leading to potential issues and crashes due to outdated results. This has been fixed.
- *(26.02.0)* In a project **EasyClassify** or **EasySegment Unsupervised**, importing a dataset with unlabeled images could lead to an error. This has been fixed.
- *(26.02.0)* When training a tool **EasyLocate**, the displayed `Best validation AP` was the `ElocatorMetrics.AveragePrecision50` instead of the `ElocatorMetrics.AveragePrecision`. This has been fixed.
- *(26.02.0)* When importing a dataset, the label color and opacity were not correctly imported. This has been fixed.
- *(26.02.0)* When importing a dataset with no image, the labels were not imported. Now, even if the dataset has no image, the labels are correctly imported.

Easy3D

- *(26.02.0)* The objects `E3DBox` were drawn 20% bigger than they should in the **3DViewer**. This has been fixed.
- *(26.02.0)* Some `E3DPlanes` were not drawn properly in the **3DViewer**. This has been fixed.
- *(26.02.0)* The spheres and the boxes were not drawn properly in the **3DViewer** when opacity was not 255. This has been fixed.
- *(26.02.0)* The wire frame was not drawn in the render sources if the render source was added after enabling the property `WireframeMode`. This has been fixed.
- *(26.02.0)* The `FillMode` of `EPointCloudToZMapConverter` was slightly different when ZMap was of the type 8 / 16 or 32f. It now always behaves as the 8 / 16 case.
- *(26.02.0)* Clicking on the button `Reset` disabled the point picking in all 3D tools in which points could be selected in the viewer. This has been fixed.
- *(26.02.0)* `EPhotometricStereoImager.ComputeMeanCurvatures` and `EPhotometricStereoImager.ComputeGaussianCurvatures` threw an exception when the parameter `GaussianBlurSigma` was set to some values. This has been fixed.
- *(26.02.0)* Saving an empty point cloud to a binary and compressed `PCD` file threw a `std` exception. This has been fixed.
- *(26.02.0)* The internal state of `EPhotometricStereoImager` was not released properly when setting or computing the calibration angles. Thus, calling `SetCalibrationAngles` or `CalibrateFromSphere` after a first call to `ComputeHeightMap` caused subsequent calls to `ComputeHeightMap` to throw an exception. The same pattern could also cause memory leaks. This has been fixed.

EasyDeepOCR

- *(26.02.0)* A crash in `ETextReader.Recognize` could occur when a read failed because the text did not match the specified topology. This has been fixed.
  - This issue could also make the **new Open eVision Studio** crash.

Python version

- *(26.02.0)* The methods taking an argument of type `ctypes.c_void_p` could fail with a `TypeError` when given an instance `ctypes.c_void_p` containing the value `0`. This has been fixed.
- *(26.02.0)* Multiple methods that could technically take or return nullable values would not be properly indicated with `type | None` in the Python stub files. This has been fixed.
  - This fix may impact users using static type checkers on their Python code but it has no impact on runtime operations.

Samples for embedded versions

- *(26.02.0)* The samples `EasyDeepLearningInferenceFFmpeg console` and `Qt` did not compile with versions of `FFmpeg` prior to 5.0. This has been fixed.

Neo License Manager

- *(26.02.0)* The UI application crashed while trying to apply an emergency license when there are no more such license available. This has been fixed.

new Open eVision Studio (preview)

- Interface
  - *(26.02.0)* The depth maps and ZMaps are now saved when clicking on the save icon of the 2D display widget.
  - *(26.02.0)* In a batching tool, the variable used to iterate on the elements of an array is not a `const` anymore.
  - *(26.02.0)* **new Open eVision Studio** could crash when a wedge gauge was connected to a wedge whose inner radius was set to 0 graphically. This has been fixed.
- Acquisition
  - *(26.02.0)* In the tool `Image file`, when setting the output port type to `Array of ×××`, it was no longer possible to deactivate the live mode if it was on. This has been fixed.
  - *(26.02.0)* In the tools `3D DepthMap file`, `3D ZMap file`, `3D Mesh file`, `3D Point Cloud file` and `Image file`, a new file was highlighted when pressing the up and down keys but the corresponding file was not loaded. This has been fixed.
- Easy
  - *(26.02.0)* In the tool `Region editor`, the width of the property grid could sometimes be influenced by the zoom factor of the displayed image. This has been fixed.
  - *(26.02.0)* In the tools `Region combiner`, `Region morphology` and `Region transform`, connecting a region that was entirely on the left or above the image (`maxX < 0 || maxY < 0`) caused a crash. This has been fixed.
- EasyColor
  - *(26.02.0)* In the tool `Color transform`, the tool was not saved correctly and the color transformation type entry was absent. This has been fixed.
- EasyImage
  - *(26.02.0)* In the tool `Transform`, transforming images with a size below 2×2 would make the application crash. Now, an error is reported.
- EasyMatch
  - *(26.02.0)* In the tool `Matcher`, the parameter `Max initial positions` had a smallest allowed value of 1, while 0 is also valid for this parameter. This has been fixed.
- EasyBarCode and EasyMatrixCode
  - *(26.02.0)* In the tools `BarCode reader` and `MatrixCode reader`, the button `Reset` did not clean the learned images. This has been fixed.
- EasyClassify
  - *(26.02.0)* In the tool `EasyClassify`, the model path was deleted when the browse prompt was closed beforehand. This has been fixed.
- Easy3D
  - *(26.02.0)* In the tools `3D Image to DepthMap converter`, `3D Image to ZMap converter`, `3D Mesh to ZMap converter`, `3D ZMap to Mesh converter` and `Image file`, the **Easy3D** namespace was not used properly in the generated code. This has been fixed.
  - *(26.02.0)* The tool `3D Transform` was generating wrong Python code when input was a point cloud. This has been fixed.
  - *(26.02.0)* The tool `3D Transform` was not handling the argument `inTransf` correctly. This has been fixed.

  - *(26.02.0)* The tool `3D Transform` was not handling ZMaps correctly. This has been fixed.
  - *(26.02.0)* In the tool `3D Photometric stereo`, the negative angles were not displayed properly in tool. This has been fixed.
  - *(26.02.0)* In the tool `3D ZMap to image converter`, some prefixes `oev` were missing in the generated Python code. This has been fixed.
- **Easy3DLaserLine**
  - *(26.02.0)* In the tool `3D Laser line object calibration`, the generated code of the tool did not take the parameter `Along positive Z-axis` into account. This has been fixed.
  - *(26.02.0)* In the tool `3D Laser line extractor`, the **Easy3D** namespace was not used properly in the generated code. This has been fixed.
- **EasySpotDetector** and **EasyDeepOCR**
  - *(26.02.0)* In the tools `Spot detector` and `Text reader`, clicking on a row partially reset the sorted column. This has been fixed.
  - *(26.02.0)* In the tool `Spot detector`, the setup view would sometimes squeeze neighboring widgets. This has been fixed.
  - *(26.02.0)* In the tool `Text reader`, sorting on the height and the angles did not work properly. 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.