Release Notes mViz 6.1

New barcode symbologies are supported.

Some `Blobs` methods have been extended to other image types.

Caution: for consistency, the color of the blobs is now specified as `Graph::FillColor` rather than `Graph::LineColor`.

Miscellaneous small improvements.

Image Processing

The operator `EqualTest` can now be applied to Gray16 or Rgb images.

Blob Analysis

You can now add whole regions or existing blobs to a `Blobs` object, to perform blob analysis on them. You can as well merge several of them, i.e. create blobs made of several independent parts. Check the methods `Blobs::AppendBlob` (2 overloads).

The method `Blobs::ToImage`, to fill an image with a given gray value or color over blob(s) can now be applied to `Gray16` or Rgb target images.

A similar `Region::ToImage` is available.

The features `BlobMass`, `BlobAverage`, `BlobDeviation`, `BlobMinimum`, `BlobMaximum` can now be processed on `Gray16` images, even if the original source image was `Gray`.

For consistency, the color of the blobs is now specified as `Graph::FillColor` rather than `Graph::LineColor`. As the default for `FillColor` is `NoColor`, a warning message will be issued if you don't set it to a visible color. This can break existing code.

Classification

The methods Classifier::PlotY and Classifier::PlotXY now allow more flexibility in the representation. Class coloring as well as dots instead of lines are now possible.

Barcode reading

The Code1DReader now supports these symbologies:

- Matrix 2 of 5
- ITF14 (GTIN-14)
- Code 11
- MSI (MSI-Plessey)

You can enable/disable them via the corresponding `Detect` flags. All are enabled by default. We remind you that it is better to enable only the symbologies that you will meet, for faster and more reliable processing.

The accuracy of the `Decodability` quality indicator has been improved, leading to better stability.

Template Matching

The `Locator::Find` method can now be applied to `Gray16`, Rgb or Rgba images. They are implicitly converted to Gray. This was already possible with `Locator::Train`.

Inspection

The SnakeGauge tool for inspection of paths and outlines now computes more features: `MaxIntrusion` and `MaxProtrusion` are the largest deviations from the model. They can be obtained globally or per section. `NumAnomalies` counts the defects longer than a specified length.

The `SnakeGauge` tool now has a method `RectangleAppend` to create a (possibly rotated) rectangular outline in a single go.

3D Inspection

Points of a cloud can now be selected by means of a 3D box. Euler angles can be specified to adjust the orientation of the box. Check the method `PointCloud::ByBoxSelect`.

The new method `PointCloud::RangeImage` can be used to generate a range image (gray value = depth) from a mesh.

Release Notes mViz 6.0

This new major release introduces the `Comparator` object, which performs defect detection by comparison to a golden template image.

The Comparator object is covered by the Inspection license, as is the Snake tool for shape control. Existing 5.x and earlier versions will have to be upgraded to 6.x.

Check the sample program `InspectCompare` and the utility `mViz Comparator`.

The Quality Indicators are now also available for the Dot Code symbology.

General

The method `Image::Limits` now returns an XY struct rather than `Site`, to account for even/odd sizes.

Image Processing

Color component swapping (RGB to BGR or conversely) is now available as a new operator. Check the method `Operator::RedBlueSwap`.

The geometric transformation methods now return a Boolean value telling if overrun occurred (i.e. some of the pixels did fall outside the source image and were interpolated by replication).

The geometric transformation methods now have an overload taking a `Region` argument which reports the areas of the destination image that were computed from corresponding source pixels rather than extrapolated due to overrun.

The method `Histogram::Normalize` now has an overload that lets you specify the mean and variance values that you want to set, rather than computing these from a source image.

The method `Operator::Binarize` can now generate a binary (Gray1 = 1 bpp) image.

Image Analysis

The method `Region::Binarize` can now process a binary image as input (Gray1 = 1 bpp).

You can now define or extend a `Region` with a rectangle specified as `Limits` or as the current `Window` of an image, or with another `Region`.

The `Region` object now has a `DrawOutline` method. In addition to the existing Draw method, it draws the external and internal contours (holes).

Blob Analysis

The `Blobs` class now has *double-thresholding* capability, detecting either the low and high intensities, or those comprised between two thresholds. Check the new method `Segment`.

The method `Segment` taking `Background/Foregound` `Gray` arguments has been withdrawn for lack of usefulness. The overload with `Rgb` arguments is kept.

The method `Blobs::FeatureAverage` has an extra argument `Average` allowing to chose between the average value or the sum.

Barcode reading

Some Quality Indicators of the `Code1DReader` could be wrong in case the last bar in a barcode was the last feature in the image, due to off-by-one error. This has been fixed.

Dot Code reading

The Unused Error Correction quality indicator could be wrong in some cases due to premature correction of the errors. This has been fixed.

The Code2DReader has a new utility function to convert raw content to the MIL-STD-130 conventions. Check the method `Code2DReader::ExpandMIL`.

The Quality Indicators are now also available for the Dot Code symbology.

Inspection

The brand new class `Comparator` is available, for intelligent image comparison. It does image registration to compensate displacements, contrast renormalization to compensate light changes and statistical tolerancing.

mViz Comparator

This new utility program is made to allow you to quickly build an inspection model for use with the Comparator objects.

Sample programs

The C++ Native sample program `InspectCompare` has been added to illustrate simple use of the Comparator. (Not available for Visual Studio 2008.)

~~mViz+~~

~~The `Comparator` has been integrated in `mViz+` in a minimalistic way. Currently, you can only reload an existing model and apply it for alignment or inspection.~~ Soon to come.

Release Notes mViz 5.6

In this release, the Region object supports morphological operations such as Dilation and Erosion as well as checking for inclusion, exclusion, and new features…

The `CharReader` (OCR) now has a font optimization capability.

Many compute-intensive functions in `Image Processing` have been optimized using the AVX/AVX2 vector instruction set, for faster execution.

General

The `Graph` attribute `FillColor` now allows to draw the closed shapes with filling. By default the setting is `NoColor`. The shapes concerned are `Dot`, `Box`, `Rectangle`, `Circle`, `Ellipse`, `Path` and `Poly` (when they are `Closed`).

CAUTION: the previous default value was `White`, so the new default can cause some shape filling (e.g. Regions) to become invisible, until you set `FillColor` to the desired value.

Image Analysis

The class Region has a few new features: `Area`, `Center`, `Mean` and `Variance`. As their names imply, they compute, respectively the area, gravity center, mean gray-level and gray-level variance inside a region.

The class Region has new methods `Dilate`, `Erode` and `Gradient` that use a `Morpho` argument to specify a structuring element. The result is a new region that it the result of the morphological operation, as if the region was a binary image.

The class `Region` now has methods to compute the (rotated) bounding rectangle and bounding circle of a region. Check the methods `Region::Rectangle` and `Region::Circle`.

The class `Region` now has a method `Region::Append`, to transfer the content of an existing region.

Blob Analysis

There is now a method `Blobs::BlobInRegionSelect` that works in a similar way as the `BlobInWindowsSelect`, but relies on a general `Region` rather than a rectangular `Window`.

You can now add blobs “manually” to a `Blobs` object, by means of the method `Blobs::AppendBlob`. The new blob is supplied as a `Region` and can then be processed as a regular blob (feature computation, selection, sorting…)

The new method `Blobs::RegionBlob` informs about the relative positions of a blob with respect to a region, using the modalities in `RangeMode` (`Inside`, `Outside`, `Onside` or `Over`). This is similar to the `WindowBlob` method.

The individual runs that form a blob can now be queried using the methods `Blobs::NumBlobRuns` and `Blobs::BlobRun`.

OCR

When trained fonts include several instances of the same characters, the font can be optimized to improve the running time by retaining an essential subset. Check the method `CharReader::Optimize`.

Barcode reading

The methods `Code1DReader`/ `Code2DReader::ExpandGS1` turn the standard string representation of a decoded barcode to a GS1 representation, with `Application Identifiers` isolated.

mViz+

There was some mess in the selection of blob features (radio buttons) in the Evaluate, Select and Sort tabs. This has been fixed.

Release Notes mViz 5.5

The release features substantial improvements of the OCR module, in order to ease the use of the predefined fonts. The sets of font images and files have been restructured.

General

The class `Status` now has two methods to implement timeout management. `Status::Timeout` starts a timer for a desired duration, while `Status::Expired` returns `True` as soon as the timeout duration has been exceeded. Note that it is the duty of the user to call the `Expired` function often enough. It is a non-blocking function, not causing an interruption.

Image Analysis

The feature `BlobFullPerimeter` has been added. It counts the number of pixels on the outline(s) of a blob, including holes. By contrast, the existing feature `BlobPerimeter` only counts the pixels of the outer contour. Hence, `BlobPerimeter ≤ BlobFullPerimeter ≤ Area`. A corresponding `Blobs::FullPerimeter` function is also available. Note that the feature `BlobPerimetricRatio` is still computed from the `BlobPerimeter`.

The method `Blobs.Point` returns the coordinates of the top-left pixel of a blob. This is useful to write a label near the blob.

OCR

New font files are available, together with the corresponding images that were used for training. For ease of use, whole character sets are included, to avoid a multiplicity of separate files.

A new property, `GlobalFilter`, is meant to select only the part of the character set that is relevant. The convention is as follows:

	Alphabetic	Lowercase	Uppercase	Digit	Special	Any
Alone	a	l	u	9	.	*
+ digits	A	L	U			
+ special	b	m	v	0		
+ digits + special	B	M	V			

A new property, `InkingCorrection`, has been added to adjust when the stroke thickness of the characters to be recognized does not match the thickness of the trained ones. This can occur when using the predefined font files.

The class `CharReader` has the new (advanced) property `OCVScoring`. It is used to assess the score of the characters even if they were specified verbatim in the filter (so don't need to be recognized). In OCR applications, these scores are not needed. In OCV applications, they are an inspection result.

The method `CharReader::GetLayoutBoxes` can now be used to retrieve the coordinates of the bounding boxes after character segmentation.

The internal spacing rules in the reader have been modified. This can imply differences in the segmentation results in the limit cases.

Gauging

The SnakeGauge now has a `Clear` method to reset it.

mViz OCR

The new features `GlobalFilter`, `InkingCorrection` and `OCRScoring` are now supported.

The use of a Window to delimit the reading area is now optional. Check the Zoom menu.

Zooming is now automatic when you load an image.

The path to the font files was not remembered correctly from one execution to the next. This has been fixed.

Release Notes mViz 5.4

mViz is now available for the ARM64 processor under Linux. In particular, it is available, on demand, for the Raspeberry Pi platform.

We remind that mViz is also available on demand for the x64 processor under Linux.

General

The class Graph was preallocating a number of GDI objects to avoid repetitive creations/deletions. Dynamic objects are now used, drastically reducing the GDI resources consumption.

When reading a palettized bitmap (BMP) image (8 bpp), the palette is honored. This complicates the determination of the image type (Rgb or Gray) when the file is read as Undefined. We have adopted the following rule: when all palette entries are achromatic (three equal components), the image is read as Gray; otherwise as Rgb.

The display of Graph::Dot with a zero size (single pixel) was off by one. This has been fixed.

The method Operator::Copy can be used for type conversions in addition to mere copying. So far, in-place type conversions (same source and destination) were not possible. This possibility has been added.

Image Analysis

The new methods Blobs::FeatureMaximumIndex/FeatureMinimumIndex return the index of the blob that has the largest/smallest value of a feature. This allows to query other features of this blob.

The features BlobFeretBox and BlobDiametralBox are now computed in such a way that Width≥Height and Angle is between 0 and a half-turn. The same convention holds for Path::Rectangle and XY::Rectangle.

The method Profile::LocalExtrema has been added to detect maxima or minima in a profile.

The methods Morpho::RegionalMaxima/Minima are controlled with a parameter MaxCount, which limits the number of points reported (the best ones). Now when the value 0 is passed, all the detected points are returned.

Calibration

Two new calibration modes have been added: BiQuadratic and Rational. By contrast with the Quadratic mode, BiQuadratic does not assume that the deformation is centered on the image and requires at least six landmarks. Rational is the combination of BiQuadratic and Perspective.

Snake

The method SnakeGauge::EdgePoint was reporting an error due to bad index checking. This has been fixed.

Code Reading

It is often the case that when printed with ink jet printers the Dot Codes are skewed due to scanning motion. The new method Code2DReader::DCCorrectGeometry can be used as a hint to the reader to improve the readability, when the skew information is available.

Due to an unfortunate change in the license management, the Demo license was only enabled for the EAN13/EAN8 symbologies. This has been fixed.

In some cases, the `Code1DReader`, with `NumDirections` set to `Auto`, could report slanted codes twice. This has been fixed.

mViz+

Due to the introduction of templated types in the 3D Module, the variable data types could be shown incorrectly in mViz+ and in the generated scripts. This has been fixed.

In the scripts, some data types were incorrectly prefixed with the namespace or misspelled in VB. This has been improved.

mViz OCR

The path to the font files was not remembered correctly from one execution to the next. This has been fixed. Zooming is now automatic.

Release Notes mViz 5.3

This release brings a major progress with the `3D inspection` module. This module includes support to represent point clouds (sets of unrelated points in space obtained from some 3D image source) and point meshes (triangulated surface).

The module supplies different geometric measurements such as dimensions, surface, volume, performs registration and surface comparison, in order to report defects with respect to a template.

Image processing

The definition of a `Region` from a Quad object was missing the first row in the case of an axis-aligned rectangle. This has been fixed.

When appending a Quad to a circular or annular `Region` of aperture 360°, pixels could be missing at the junction, due to numerical errors. This has been fixed.

You can now define the inside of an ellipse as a `Region`. Check the method `Region::EllipseAppend`.

The operations on `Regions`, namely `Invert`, `Intersect`, `Unite` and `Subtract` were producing wrong, unpredictable effects when done in-place (one of the source region arguments being the same as the region instance). This is fixed. Hence repeated operation on the same region (for aggregation) is now possible.

The classes `Path` and `Poly` support new geometric methods to obtain the tightest bounding rectangle and bounding circles. Check the methods `Path::Rectangle`, `Path::Circle`, `Poly::Rectangle` and `Poly::Circle`.

Image Analysis

The method `Blobs::Segment` on a grayscale image taking a `Region` argument (`Mask`) could cause crashes in case the limits of the region exceeded the image vertically (ordinate `<0` or `≥Height`). This has been fixed.

The `Path` and `Poly` objects now have methods to compare a closed shape to a known rectangle or a known ellipse, by computing the average distance between the vertices and the shape. Check the methods `RectangleDeviation` and `EllipseDeviation`. These can be used with the results of ellipse fitting or bounding rectangle computation.

Code Reading

In some cases, error correction of the Data Matrix codes was behaving incorrectly, giving incoherent decodes. This has been fixed.

The reading rate of the Dot Codes has been significantly improved.

Character Reading

There was a wrong handling of the `Constrast` parameter in the mode `WhiteOnBlack`, preventing correct character rating. This has been fixed.

3D inspection

Check the new chapter "3D Inspection" in the manual.

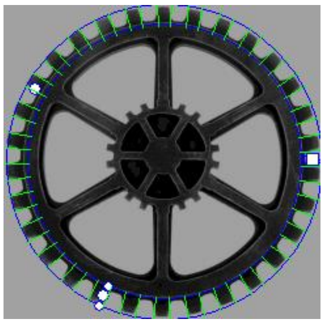
Template matching

The search can now be performed in a region of interest of arbitrary shape. Check the methods `Locator::SearchMask`. The mask needs only be set once before searches are performed.

The methods `Locator::Draw` and `Locator::Location` now take an extra `Depth` argument, which is useful mainly when `StopDepth` differs from `0`. The default value, `-1`, ensures that the location at `StopDepth` is reported. The value `0` corresponds to full scale.

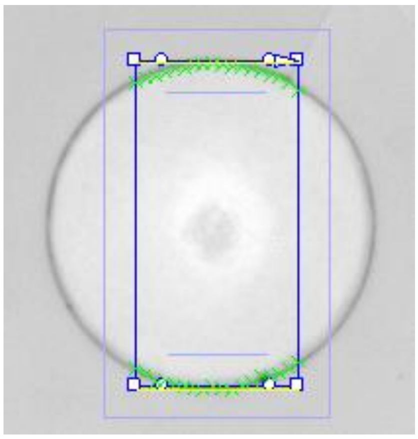
Gauging

The `EdgeArc` gauge is now able to detect edges across the circumference rather than along it. This is similar to what an `EdgePoint` gauge can do, but on a curved segment.



Detection along the circle

The EdgeRectangle can now act as a caliper, i.e. measure the distance between two edges in a single go. This is obtained by setting the parameters `FourSides=false` and `FittingSide=Before`.



Caliper function

mViz OCR

The utility `mViz OCR` has a new mode of operation. After characters have been detected, it tries to read them using a set of font files, in order to guess the effective font (if already known).

mViz 3D

The new utility `mViz 3D` is available to test and try the 3D module capabilities as well as to generate inspection models.

Release Notes mViz 5.2

New in this release:

Explicit support for Visual Studio 2022 has been added (precompiled libraries and sample programs).

The .NET wrapper is now compiled for the framework version 6.0, in addition to 3.5, 4.0 and 4.6.

The decoding capability of the `Code2DReader` has been significantly enhanced. It is able to decode the lacunary codes that could be generated when following the pre-4.0 standard (though the latter are obsolete).

The `EdgeEllipse` class has been added to the gauging tools.

General

The attribute `Graph::LineThickness` (in pixels) has been added. The default is `1`, as before.

`Graph::EllipseArc` has been added to draw a partial ellipse.

Image processing

Now all image processing filters (transforming an image into another image) can be applied in-place, i.e. with the destination same as the source.

`Region::Fill` could result in wrong results or access violation in case that the `Region` exceeded the source image limits. This has been fixed so that only the pixels that fall inside both the source and destination images (or `Window`) are copied.

The functions `Histogram::Noise` and `Histogram::ShowNoise` now allow to choose between two filter sizes, the larger being more accurate.

Dot Code Reading

The property `Code2DReader::ObsoleteDotCode` allows working around a flaw in the standard before version 4.0, such that lateral rows with no dot at all are possible. The Reader will process such codes correctly. For newer codes that comply with AIM ISS DotCode Symbology Specification 4.0, leave this property as `false`.

The property `Code2DReader::QuietZoneClutter` lets the reader deal with spurious stains or debris in the quiet zone area. If the periphery of the code is clean, you can disable this feature.

The reading capability of the Dot Codes has been improved.

Gauging

An `EdgeEllipse` gauge has been added. It works in a way similar to the `EdgeArc`, and allows two axis of different length and arbitrary direction. It can be useful to measure circles viewed in perspective, or real elliptic shapes, open or closed.

In the drawing of the gauge handles, the beginning handles are now drawn square rather than round, to distinguish. This is purely cosmetic.

The property `EdgeArc::Circle` has been renamed `EdgeArc::Closed`, for consistency. This can impact code that uses it, but the functionality is unchanged.

mViz+

Support for the `EdgeEllipse` has been added.

Some variable declarations could be duplicated in the generated scripts. This has been cured.

Release Notes mViz 5.1

New in this release:

The selection of the instances in pattern matching is now governed with a contrast criterion in addition to the matching score. This is useful to avoid false occurrences found in dark areas.

General

A function `Graph::Rectangle` has been added to draw a rotated rectangle. We recall that an upright rectangle can be drawn with `Graph::Box`.

The size of an image can now be obtained as a pair `Site Image::Size()` rather than separate `Width`, `Height` properties.

The `Vertex` argument of `ArcPath::Draw` (drawing of a curvilinear path) has been changed from Boolean to integer to allow specifying a marker size. This could break existing `.NET` code.

Image processing

The argument `Connexity8` of `Morpho::RegionalMaxima` was not honored when `false`. This has been fixed.

The method `Path::Smooth` was not always handling closed paths correctly near the closing. This has been fixed.

`EdgeMap::Draw` now allows to specify the size of the vertices rather than presence/absence, following the size convention of `Graph::Dot`. Caution: the argument `EndpointSize` was previously a Boolean type.

The method `Operator::Convert` for color system conversion can now be used in-place (output image same as input).

Dot Code Reading

Decoding of *rectangular* *Data Matrix* codes could sometimes fail due to bad internal initialization. This has been fixed, and the decoding rate is improved. Square codes are not impacted.

The parameter `StartLevel` can now be set to the value `-2`. This magnifies the image by a factor 4 (while -1 corresponds to a factor 2). This feature should only be used on difficult tinny codes, as it will significantly increase the running time.

Template matching

The method `Locator::Train` with a `Region` mask argument was assuming that the region origin was (0, 0). The method now takes an extra argument `SrcWindow`. When true, the origin is the origin of the `Src` image (in case it has a Window). As the default is false, existing C++ code is not impacted. .NET code requires the new argument.

The property `Locator::MinContrast` is now exposed to allow filtering out too dark instances. The default value is `0.25`, meaning that reducing the ambient light by a factor four will result in the disappearance of all instances. In such situation, the minimum contrast can be lowered.

As a consequence, 1) the `Pose` information now contains the measured contrast as member `Contrast` and 2) a new overload of `Locator::Draw` allows enabling the display of the numerical values of the scores and contrasts.

mViz Code2D

The parameter `StartLevel` (used to optimize the search for tiny or huge codes) is no more exposed as a variable range. This was useless as all levels above it are tried anyway.

Release Notes mViz 5.0

New in this release:

- Improved omnidirectional barcode reading;
- Improved line and circle fitting, and ellipse fitting;
- The `Locator` can be trained with an "external" edge map;
- Miscellaneous fixes and improvements.

As of `mViz 5.0`, licenses for `5.x` are required. Upgrades from lower versions can be obtained.

The new module `Inspection` requires a specific license. Anyway, it is also included in the `Full Set`.

General

The function `Graph::ZoomToWindow` was ignoring the presence of scrollbars, sometimes causing part of an image to remain hidden. This bebavior has been changed.

Saving an image with a `.jpg` or `.jpeg` extension resulted in the file being a BMP format internally, instead. This has been fixed.

The properties `Width` and `Height` of the `Region` objects have been discarded because of the cost of the (unnecessary) updates. These values can now be explicitly queried via the method `Limits Box(int Margin= 0)`.

`Graph::Offset` now has an overload taking a `Site` argument instead of separate coordinates.

Image analysis

Methods `Path::EllipseFit` and `Poly::EllipseFit` have been added. They allow a best fit of an ellipse to a `Path` or `Poly`. In the same way as the similar `LineFit` and `CircleFit` methods, they support `Accurate` (outlier-free data) and `Robust` modes.

The function `Path::Ellipse` was conceptually wrong and we decided to remove it. It is now replaced by `Path::FilledEllipse`, that has a different behavior. If that causes an issue for you, please contact the technical support.

The method `EdgeMap::GradientMaxima` has a new parameter `UpperLimit`, that allows to suppress the high gradient values (the low values are suppressed below the parameter `Noise`).

The methods `Draw` of the `Pyramid` object now draws the images of the various levels in a non-overlapping way. This behavior can be changed by means of the `Overlapping` argument.

The required offset for a given pyramid level can be queried with `Pyramid::DrawingOffset`.

Gauging

The parameter `Size` of `EdgeRectangle::Locate` was incorrectly passed, resulting in wrong gauge dimensions. This has been fixed.

The line segments suffered from a wrong reference direction, causing the polarities to be incorrectly handled. This has been fixed.

Template matching

The Locator can now be trained with an edge map image rather than a plain grayscale. This allows to feed the result of a gradient filter or the output of the `EdgeMap::GradientMaxima` or `EdgeMap::StrongEdges` methods. The map must be non-maxima suppressed (thin edges).

Bar Code Reading

The `Code1DReader` now supports reading in arbitrary directions without having to adjust `NumDirections`. As a benefit, the running time does not increase with `NumDirections`. To obtain this effect set `NumDirections` to the `Auto` value, as is now done by default. CAUTION: if existing code relies on the previous default value 2, consider choosing between `Auto` and 2.

OCR

The property `CharReader::MinimumContrast` has been added. It is helpful to discard dark features that could be taken for characters.

Dot Code Reading

The property `Code2DReader::AccurateEdges` is no more activated by default, as this is not always necessary.

Inspection

The methods `EdgePoint` and `RibbonPoint` have been added to the `SnakeGauge`. They allow you retrieve the coordinates of the points found during inspection.

Caution, for consistency, `SnakeGauge::AssessEdges` has been renamed `SnakeGauge::AssessEdge`. Existing code will be impacted.

The folder `Images` now contains the files `Gasket Ribbon.png` and `.sng` that are used as demos for the ribbon inspection mode.

There were several inconsistencies in the GUI of the utility `mViz Snake`. They have been fixed.

The Help information was wrong. This has been fixed.

Interaction with zoom on could cause wrong behavior. This has been fixed.

mViz+

The class properties with an enumerated type are now echoed as the relevant enum value rather than a cast integer.