CAUTION:

The current installer cannot install the 64 bits dongle driver. If you plan to develop 64 bits applications, please download the file from this link (despite the warning, the link is safe) and copy it to the Windows\System32 folder.

Release Notes mViz 3.6.3

The is the last release before the next main major version, 4.0.

The `Code2DReader` now has multiple codes reading capability.

General

Precompiled libraries for Visual Studio 2013 and 2015 are now available. The Visual Studio 2005 versions have been discarded (contact techsupport@visionforvision.eu in case of need).

The .NET library for use with the framework version 4.0 is now available.

The display of images was relying on the native GDI functions in Windows. In the case of zoom out, these functions use decimation, resulting in strong aliasing and possible disappearance of lines and thin features. The display now relies on correct image resizing performed by mViz, resulting in much better rendering. Anyway, you can revert to the original behavior by setting `Graph::SmoothZoom= false`.

Copy of a `Gray1` image could cause invalid memory accesses. This has been fixed.

A graphical function has been added to draw polylines/polygons defined by an array of XY points. See `Graph::Poly`.

Blob Analysis

The segmentation functions of the `Blobs` class are now taking a `MaximumArea` argument in addition to `MinimumArea`, to implicitly ignore large blobs. By default the value is `MaxInt`, causing no blob rejection. Existing C++ code won't be impacted. .NET code will require to pass the new argument (use `Int32.MaxValue` to have no effect).

The member Blobs::ByFeatureSort was not performing correctly on Float features. This has been fixed.

A new statistical feature parameter has been added: the mode of the distribution, i.e. the most frequent value. Check the method Blobs::FeatureMode.

A ClearBlobs method has been added to reset the Blobs object.

The Gray16 image type is now supported for segmentation.

Image processing

The Geometry::DownSample function has an additional Filter argument to disable low-pass filtering (that reduces aliasing). When the argument is set to false, pure decimation is performed instead. As the default value is true, existing C++ code won't be impacted.

The adaptive thresholding function Operator::Binarize(Size, Noise, …) couldn't be executed in-place. This is now possible.

The morphological filters (class Morpho) now support the Gray16 images.

Adaptive binarization of the Gray16 images is now supported (class Operator).

The class Pyramid now has a Double method to resample an image with resolution doubling.

Code Reading

Multiple DataMatrix or QR codes can be detected and decoded simultaneously, without specifying individual ROIs. This is achieved by setting the property MaxDecoded to a higher value than the default 1 before calling Decode. Then NumDecoded indicates how many codes ware found and you can retrieve their characteristics one by one using the method Choose.

The method Code2DReader has two new Boolean arguments: All, that allows selecting to draw only the currently chosen code or all codes simultaneously, and HighlightCurrent which represents differently the current code.

CAUTION: the argument Errors has been withdrawn, as this functionality is obsolete.

The drawing options All and HighlightCurrent are also available for the barcode reader.

The performance of the EAN128 reader has been improved.

The performance of the DataMatrix reader regarding reading of rectangular codes has been improved. Remember that you need to explicitly select the rectangular codes in the From/To range.

The CellSize was not computed for the QR codes. This has been fixed.

The reader now allows to manually adjust the binarization threshold used to detect the code, via the new property Threshold. This only works when AdaptiveThreshold is false. By default the threshold is set to Auto which selects the value automatically. Anyway, manual adjustement is discouraged and must be avoided when computing the Quality Indicators.

The StringLength after decoding a QR code did account for the final null byte. This has been changed and the value is now one unit smaller.

Template matching

The Locator now has a method Clear to reset the locations list.

Character Reading

The reader now allows to manually adjust the binarization threshold used to detect the characters, via the new property `Threshold`. This only works when `LocalThreshold` is `false`. By default threshold is set to `Auto` which selects the value automatically. Anyway, manual adjustement is discouraged.

The accuracy of the segmentation of touching characters has been improved.

Gauging

The Line/Arc/Rectangle gauges now return the number of points that were detected and those that were effectively used for (robust) fitting. These can serve as an additional assessment of the quality of the fit. Check the properties `NumDetectedPoints` and `NumFittedPoints`.

The `Distance` arrows could be wrongly drawn between `EdgeLines`. This has been fixed.

mVizNET

A method `Image::ToImage` has been added to convert an `mVizNET` image to a .NET Bitmap object. The types supported are `Gray1`, `Gray`, `Gray16`, `Rgb`, `Rgba`.

The attribute `TextFont` for text drawing needed to be passed as an array of bytes instead of a plain `String`. This has been fixed.

mViz+

In the `Blob Analysis` tabs, one can now enable/disable the display of the unselected blobs by means of the “View all” checkbox.

In `Blob Analysis`, Features Distribution tab, the `Int` and `Float` feature quantiles have been regrouped in a single function `FeatureQuantile` returning a `double` value. Existing code will break.

mViz OCR

The application was trying to save the OCR settings in a font file to restore them at the next execution. The file was saved next to the application. Under Windows 10, this causes an access denial error. The file is now saved in the temporary user folder instead.

Release Notes mViz 3.6.2

This version was not released.

Release Notes mViz 3.6.1

This is a maintenance release that mainly fixes issue in the `Locator`.

General

The class `Limits`, that handles axis-aligned bounding boxes has been enhanced with

- A method `Center` to obtain the center coordinates,
- A method `Area` to compute the box area, in pixels;
- Two methods `Include` to extend the box by incorporating additional points or boxes;
- A method `Intersect` to restrict the box to its intersection with another.

Image analysis

The class `Path` has a method to compute the convex hull of a simple polygon. This method has been augmented with a Boolean flag to allow processing an arbitrary point cloud (or a crossed polygon). By default, the polygon is assumed simple so that the previous behavior is unchanged.

Histogram equalization can now be done in an adaptive way, i.e. using a sliding histogram of a fixed size. Check the overloads of `Operator::Equalize`.

Template matching

Due to a typo, the `Locator` was returning extra random instances when the number of valid instances found was lower than the requested number (`MaxLocations`). This has been fixed.

Code Reading

The tolerances on bar thicknesses have been loosened to deal with non-standard Code 39 symbols.

Release Notes mViz 3.6.0

This is an ordinary intermediate release that features histogram, CodeReader and OCR improvements, among others.

General

The `Window` method (used to define an ROI) now returns a Boolean value to indicate if the window wholly fits in the image or had to be clipped.

The file format Portable Bitmap (`*.pbm`) is now supported. It conveys binary images (type `Gray1`).

The angle parameters stored in external files (`Geometry`, `Locator`, `CharReader`, `ShapeMatcher`) had to be saved/loaded using the same `AngleUnit`. This is no more required, upon loading, the parameters are converted to the current unit.

Image processing

The morphological thinning function has been improved in several respects. It is now split in two functions, `Thin` and `Thick`, which supersedes the effect of the `WhiteOnBlack` argument. It has a new argument, `MaxLayers`, which allows stopping the erosion process prematurely (and keep the large objects). Last but not least, it is much faster on large shapes.

The method `Histogram::ClearBins` has been added to allow resetting a user-computed histogram (the standard histogram computation functions already have a `Clear` option).

The method `Geometry::Unwind` wasn't working properly on `Rgb` and `Gray16` images because of wrong coordinate computations. This has been fixed.

The pseudo-coloring LUT obtained with the method `Profile::PseudoColor` can now be defined with direct or inverse polarity (black-to-white or black-to-white mapped to blue-green-red).

The method `Path::ConvexHull` was restricted to paths made of connected pixels (such as the result of the `Follow` operation). This restriction has been lifted and polygons are also handled. Anyway, the outlines must be simple (non self-crossing).

Methods have been added to the Histogram class to support the Maximum Entropy (Kapur, Sahoo & Wong) and ISODATA (Ridler & Calvard, Velasco) methods of automatic threshold computation. Check the `Histogram::EntropyThreshold` and `Histogram::Isodata` methods. (Once the threshold is obtained, pass it to the `Operator::Binarize` method.)

Code Reading

The DataMatrix reader (Code2DReader) has been improved to detect the codes in more complex environments.

Some improvements have been brought to the Quality Indicators for better compliance with the standard.

A new mode of operation is available: Code2DReader::AdaptiveThreshold, which is true by default. It allows to handle codes in adverse illumination conditions. Anyway, for compliance with the ISO standard, this mode should be disabled when computing the Quality Indicators.

Character Reading

When reading text in the VariableWidth mode, matching to characters of a very different width (by a factor more than two) is no more attempted. This improves the reading rate of the narrow characters such as i or 1.

The parameters MinAngle/MaxAngle are now stored in the font file. When loading existing files of a previous version, the current setting are left unchanged.

Two special modes of operation have been added to deal with difficult segmentation cases : Masking and NumRows.

Gauging

The direction arrows now have a zoom-independent size for better viewing on large images.

The method EdgeRectangle::Locate didn’t take into account the Width argument (though it could be adjusted as a class member). This has been fixed.

Due to incoherencies, the Center argument (passed to Locate) and the Center property in the EdgeRectangle has been renamed Middle. This point is now the center of the gauge, i.e. the symmetry center of the rectangle. For that reason, the FittedCenter result has been renamed FitterMiddle. Existing code will be impacted.

Release Notes mViz 3.5.0

This is a minor release that features a new sample program for the ShapeMatcher and new font files for OCR.

General

When reading a PNG image, implicit conversion to the Gray type is possible.

Image processing

The Path object has a new method PolygonToRaster, which turns a polygonal outline to a digital contour, i.e. such that all vertices are connected.

It also has four new methods to compute path features : Mass, Centroid, Ellipse and Histogram. The first computes the sum of the gray-levels along the path, the second is the gravity center of the vertices, weighted by their gray level. The third is the ellipse of inertia and the last computes the histogram of the gray values.

Gauging

The direction arrows now have a zoom-independent size for better viewing on large images.

Character Reading

The folder `Image\Fonts` contains a number of pre-learnt font files for a few standard fonts : OCR-A, OCR-B and SEMI (used in the semiconductor industry) ; Arial font files are also available and feature variable-width characters. The files contain either uppercase letters, lowercase letters, digits or all of these plus a few punctuation symbols.

Template matching

There is a new sample program named `MatchShape`. It demonstrates theuse of the ShapeMatcher tool, which is used to recognize and locate objects known by their outlines.

The `ShapeMatcher::Draw` function now takes an extra argument `Similarity` to enable the display of the score along with the class label.

Release Notes mViz 3.4.0

General

This new release has an important new feature named the `ShapeMatcher`. This is a geometric template matching tool that will recognize objects based on their outline shape and determine their accurate position, orientation and size.

Another welcome improvement is the support for rectangular Data Matrix codes and a more complete set of quality indicators. Compliance with the standard has also been improved.

The indexing scheme in the `Blobs` analyzer has been streamlined for better uniformity.

Blob Analysis

The `Region` and `Blobs` classes have a new method to modify a shape: `Convexify` replaces the shape by its convex hull.

The resulting blobs/regions can be used for further processing, such as feature computation.

The feature `PerimetricRatio` has been added. It is a good indicator of the circularity of an object. For a perfect circle, the value is 1, and is smaller for elongated/sinuous shapes.

The feature `ConvexPerimeter` has been added. It is the length of the convex hull of the blob, and indication of its size.

The feature `ConvexPerimetricRatio` has been added. It is a good indicator of the concavity of an object. For a convex shape, the value is 1, and is smaller for a concave one.

The feature `ConvexAreaRatio` has been added. It is another indicator of the concavity of an object. For a convex shape, the value is 1, and is smaller for a concave one.

The method `DrawBlob` has as additional flag to display the `Class` index instead of the blob Index when the `Label` display is enabled.

All the `BlobIndex` arguments are now handled equivalently and they all refer to the blobs indirectly, i.e. taking the selection or sorting into account (previously the individual feature computation functions were using the straight indexes).

Code that uses one of the « Individual blobs processing» approach or the « Collective processing » approach alone should not be impacted. Code mixing the two modes might. In case of a doubt, please contact the Technical Support.

Image processing

mViz now has shading correction capability on color images too (in addition to grayscale w/o extended precision). The methods `Operator::Add/Multiply/Correct` for shading correction can now be applied to an Rgb image with either Gray (achromatic correction) or Rgb (full correction) references.

Binary thinning of a binary image is now available. This reduces shapes to their skeleton, without altering connectivity. Check the `Morpho::Thin` method.

Code Reading

The `Code2DReader` now supports the rectangular codes. To enable the reading, you need to adjust the `SizeFrom/SizeTo` range to include what you need among `Size8x18`, `Size8x32`, `Size12x26`, `Size12x36`, `Size16x36`, `Size16x48`.

The following quality indicator grades have been added :

- `GridNonUniformity`,
- `Modulation`,
- `ReflectanceMargin`,
- `FixedPatternDamage`.

The optional `ContrastUniformity` parameter is also available.

Note that the grades have been reordered to :

> `Overall Grade`, `Symbol Contrast`, `Print Growth` (`Horizontal` and `Vertical`), `Axial Non Uniformity`, `Grid Non Uniformity`, `Unused Error`, `Modulation`, `Reflectance Margin`, `Fixed Pattern Damage`.

A property has been added to improve the accuracy of the outline fitting : `AccurateEdges`. This mode is useful in case you want to rely on the position of the corners for guidance, or for more accurate computation of the quality indicators.

Another property has been added to allow speeding-up the processing in case of high resolution images, such that the cells are big (like 10 pixels in size of more). When this parameter, called `StartLevel`, is increased, the image is reduced for faster processing.

The Quality Indicator `PrintGrowth` was based on default values `Minimum=0/Nominal=1/Maximum=2`. This has been changed to `0.35/0.5/0.65` to better match the standard. Existing code can be impacted, contact us if necessary.

Template matching

Check the new class `ShapeMatcher`, and the **Shape Matching** chapter in the manual.

The locator now has a method to let you normalize the pose of the located instances by canceling the rotation and scaling so that they can be overlapped on the trained image. A new image showing the instance in a standard position results. Check the method `Locator::Register`.

Release Notes mViz 3.3.0

General

The release introduces a new classifier tool, which will allow you to perform pattern recognition, i.e. identify objects by comparison with other, pre-trained objects, by means of numerical features. Its use is straightforward and convenient. Check the class `Classifier` and the sample program `Classify`, or the **Classification** chapter in the manual.

Image processing

The classes Path and Poly (polygons with integer or floating-point coordinates) have new methods PointInPath/Polygon and Overlapping that implment geometrical tests : is a given point inside a polygon, do two polygons (partially) overlap ?

Blob Analysis

The feature FeretRectangle has been renamed FeretBox for consistency. This can break exisiting code that uses this feature.

A new set of features is available. Similarly to the FeretBox, which is a tight rectangle around the blob, the DiametralBox is a rectangle as large as the blob diameter (longest distance between blob pixels). The CenterX/CenterY/Width/Height/Angle parameters are reported.

Release Notes mViz 3.2.1

General

The method Image::Read now assigns the image type Gray by default, instead of Undefined, as grayscale has “first-class” support in mViz. So loaded images are implicitly converted to grayscale. To keep the native image type, explicitly pass Undefined.

Blob Analysis

You can now compute the covariance and correlation between two features computed on a selection of blobs.

Image processing

It is now possible to fill the whole interior of a region, copying a constant value or the content of another image to an image. Check the methods Region::Fill.

The SeedFill function takes an additional Tolerance parameter that allows filling the pixels with a gray value in a range instead of an exact value.

The Histogram class supports two new methods to control saturation for contrast/brightness adjustment. The method Saturation tells the fraction of the pixels which are saturated, i.e. too dark or too light to be in the possible range. The method ShowSaturation displays the image with the saturated pixels painted blue or red. These methods are available for grayscale and color images.

The Path and Poly classes supply new Insert and Delete methods to allow interactive polyline/polygon edition.

The method Geometry::UpSample now allows to enlarge an image without interpolation, giving a pixelated effect.

Character Reading

The CharReader has a new member Dotted which helps reading dot printed characters that appear as non-touching dots. When you increase the value of this property, a morphological operation is applied to improve the connectivity of the dots.

The reader now allows you to specify yourself where the characters should be found, so that you can bypass the segmentation process. Check the methods CharReader::PresetCharBoxes/PresetCharBox.

Gauging

The methods `Detect` of all gauges now return a Boolean value which indicates if the detection succeeded. In rare cases, there can be too few points for a measurement to be possible.

The `Line`, `Arc` and `Rectangle` gauges now have the `Oblique` property set to `true` by default, to avoid unintuitive effects. This can affect existing applications. In case of doubt, you can reset the property to `false`.

Template matching

Training with a mask defined as a region is now supported. Check the overloads of `Locator::Train`.

mViz+

The interactive SeedFill operation wasn't working because of bad seed point coordinate initialization. This has been fixed.

The mViz OCR utility for font training now has a zoom feature to handle large images.

The script generator was inserting an unexpected Window() call for every image. This has been fixed.

Some of the image statistics functions were erroneously expecting an histogram input because of a confusion between overloads. This has been fixed.

Release Notes mViz 3.2.0

The main feature of this release is the addition of an automatic thresholding mode to several image processing operations.

General

The BMP files with a recent header (BITMAPVxHEADER) as produced by some third-party programs were not supported. They are now.

Blob Analysis

The segmentation methods, namely `Region::Binarize` and `Blobs::Segment` now accept the symbolic value `Auto` for the `Threshold` argument. The functions return the selected value of the threshold. The previous automatic methods taking no threshold argument have been kept for compatibility, anyway.

You can now enumerate all the runs (horizontal segments) in a region with the method `Region::Run`. There are `NumRuns` in total.

Two methods that handle handle the contours of the blobs: `Blobs::DrawOutline` and `Blobs::Outline` have been simplified. They don't require the parameters that were used during segmentation anymore. This can break code that uses these methods.

Image processing

The watershed segmentation function supports a new modality: instead of an image with black regions/white edges, it can fill the regions with the average gray-level in the original image. Check the overloads of `Morpho::Watershed`.

The method `Geometry::Upsample` used to enlarge an `Rgb` image was causing invalid memory accesses because of bad format handling. This has been fixed.

The binarization methods, namely Operator::Binarize, now accepts the symbolic value Auto for the Threshold argument. The functions return the selected value of the threshold. The previous automatic methods taking no threshold argument have been kept for compatibility, anyway.

The methods GradientMaxima, StrongEdges, Follow and Vectorize in the class EdgeMap are now able to determine the Noise level automatically. Just pass Auto for this parameter. The function returns the value computed automatically.

The same holds for the Morpho::Watershed methods. Note: the LowThreshold parameter has been renamed Noise.

It is now possible to compute a distance map from a binary image. See the method Morpho::Distance. Different variants are available.

A path can now be vectorized (turned to longer line segments) with the method Path::Vectorize.

The filter Kernel::BoxHighPass now has an additional Gain argument that allows increasing the contrast.

The class Profile now has methods to compute the range of gray-levels; like the variance, this is a measure of spread. Check the methods RowRange, ColumnRange and LineRange. All the Line… methods now support the Gray, Gray16 and Rgb pixel types.

Code Reading

The method Draw of the Code2DReader was erroneously displaying Codabar when the Type flag was set. This has been fixed.

The manual sections about code reading have been strongly reworked and extended.

mViz+

The parameters that allow an Auto value now have a small button on the side to reset the field to Auto.

In blob analysis, the blobs cannot be drawn as just an outline instead of being filled. There is a checkbox on the blobs dialog box.

Release Notes mViz 3.1.1

This release introduces improvements in the Locator tool (template matching).

Image processing

The Kernel::GaussLowPass filter was operating abnormally because of an accidental code change. This has been fixed.

Blob Analysis

You can compute the median gray value among the pixels of a blob. This is available through the BlobMedian feature (Int).

Template matching

The training of the template can now be done with a **circular** window instead of a rectangular one. This is beneficial when full rotation is allowed, as the corners of the rectangular window could be blocked by the image boundaries. The method Locator::Train has a new Boolean argument that allows to switch between the two modalities. For existing .NET code, pass false after the Image argument to keep the previous behavior, or true to work with circles.

Due to a flaw in the gross location phase, matching could fail with some combinations of the scaling/rotation ranges. This has been fixed.

mViz+

The function `Path::Follow` wasn't working properly because of wrong parameter passing. This has been fixed.

Release Notes mViz 3.1.0

The main change in this release is the support for the .NET framework 3.5 in addition to 2.0. Two separate folders, NET2 and NET35 are now available. The .NET sample programs have been upgraded to 3.5.

General

A new macro `DoubleBuffer` is available in `mVizNET.h`, to allow enabling of double buffered display for a Windows Forms item (C++). Pass the item type and item pointer.

For consistency, the method `Graph::Cross` that draws a cross marker uses the `DotColor` instead of the `LineColor`. Existing graphics will be influenced.

Image processing

The class `Pyramid`, for multiscale processing, has four new functions to compute the average, standard deviation, mid-range or range of an image, for square `N` by `N` tiles. Check the methods `Pyramid::TileMean/TileDeviation/TileMidRange/Range`.

The `Morpho::Watershed` method now has an extra argument `Binarize` that lets you choose between the region edges drawn in white or in the original gray value.

Blob Analysis

The methods `Blobs::DrawEllipse` and `Draw::Ellipsoid` were drawing ellipses twice too large because of a wrong update in GDI code. This has been fixed.

Code Reading

The symbology `Codabar` (Pitney Bowes Corp) is now supported.

The `Code39Extended` could fail to be interpreted as `Extended`, and a plain `Code39` was reported instead. This has been fixed.

The `DataMatrix` of sizes `104x104` and `120x120` were not decoded successfully because of a typo in tables. This has been fixed.

The QR codes of some large sizes could not be decoded because of finder cells misalignment. This has been fixed.

Release Notes mViz 3.0.4

This is a minor release with a few changes and fixes.

General

Images can now be saved under the TIFF and PNG formats (in addition to BMP, JPG, PGM and PPM). It suffices to pass a filename with extension "`.tif`" or "`.png`" to the function `Image::Write`. Image types Gray1, `Gray`, Rgb and Rgba are supported.

Image processing

Several filtering functions now support the `Gray16` data type: all variants of `Kernel::Gradient/Sobel/Prewitt`, `Kernel::Bilateral` and `Morpho::Median/Rank`. The function output is also of the type `Gray16`.

Two morphological operations have been added: the external (dilation minus original) and internal (original minus erosion) gradients, in addition to the existing gradient (dilation minus erosion). They are obtained with a new `Type` argument to the method `Morpho::Gradient`, which can take the value `StandardGradient` (the default), `ExternalGradient` or `InternalGradient`. Existing .NET code will require to pass this extra argument.

The class `Profile` supports initialization of lookup tables of new types, namely logarithmic, exponential and gamma. Check the methods `Profile::Logarithm` and `Profile::Gamma`.

Geometric Calibration

The `Geometry::Read/write` were not handling all calibration parameters, causing a failure to restore the original values. This has been fixed. To ensure proper working, the calibration files should be re-created.

Template Matching

When drawing the template instance(s) with `Locator::Draw` and `StopDepth>0`, the overlay was improperly positioned, even though the pose data was correct. This has been fixed.

Limitations on the template size have been removed.

Code Reading

The position parameters in the Code1DReader, namely `X`, `Y`, `Width`, `Height` and `Angle` were not updated after a read (anyway the `Draw` function behaved normally). This has been fixed.

mViz+

`mViz+ 3.0.2` or `3.0.3` could fail to launch on a machine that never ran an older version before, because of a missing Registry parameter. This has been fixed.

Release Notes mViz 3.0.3

mViz+, a powerful prototyping environment, is now a standard component of the mViz SDK. It will let you try the functions, create processing sequences and see the effect of parameters in real-time. It will also generate the corresponding source code in your favorite programming language.

General

Images can now be saved under the JPEG lossy format. It suffices to pass a filename with extension "`.jpg`" to the function `Image::Write`. The compression quality can be adjusted using the property `Status::JpegQuality`, in range [0..100]. Images types `Gray`, `Rgb` and `Rgba` are supported.

Some PGM or PPM files could fail reading because of mis-decoding of the header. This has been fixed.

Writing PGM or PPM images (binary format, P5 or P6 signature) is now possible. Just specify the desired file extension.

Image processing

A new color image binarization modality is available in the class `Operator`. It is based on the distance to a reference (foreground) color, with tolerances. A soft binarization variant is also available. Check the `Operator::Binarize` overloads.

The class `Profile` now supports extraction of oblique profiles. Check the methods `Line Get/Add/Variance/Minimize/Maximize` that take two endpoints and a Breadth parameter as arguments.

Blob Analysis

The class Blobs now supports a segmentation method using adaptive instead of global thresholding. Check the overloads of Blobs::Segment.

The class `Blobs` supports segmentation methods that are limited to the inside of a mask `Region`. Check the overloads of `Blobs::Segment`.

Code Reading

The `Code1DReader` now supports the `UPC-A` symbology, a subset of EAN13. When both symbologies are enabled and the first digit is a '`0`', `UPC-A` takes precedence.

The overlay drawing of some codes (EAN13, EAN8) was wrong on inverted codes (right-to-left, bottom-up). This has been fixed.

The policy for handling of the barcodes has changed: as of now, the `CheckQuietZone` property applies to all symbologies (and not just `Pharmacode`). It allows to deal with nonstandard barcodes with polluted or too narrow quite zone. As the default value is true, existing code shouldn't be impacted.

The decoding of Data Matrix (`Code2DReader::Decode`) including special codes `Macro05`, `Macro06` or `ReaderProgramming` could cause buffer overflow because of improper handling. This has been fixed.

Gauging

Under certain circumstances, the `EdgeArc` could report a few wrong points in the Nth mode. These points usually remained unnoticed because of the outlier rejection mechanism. Anyway, they don’t occur anymore.

Calibration

The method `Geometry::CalibrateTarget` takes two new arguments, `OriginColumn/Row`, to specify where the origin point (`World(0,0)`) is placed. Using integer coordinates maps to the center of a dot/square on the target; half-integers are in-between. This can break existing .NET code.

Correspondingly, the method `Geometry::DrawAnchors` takes a new Boolean argument `Origin` to enable the representation of the origin point.

mViz+

New clipboard operations are possible in mViz+. Right clicking on an image allows you to copy it wholly, or to copy the color value of the pixel under the cursor. This color value can be pasted as an `Rgb` constant in the operations that use one (in particular, color binarization). The paste is made by right-clicking on the color tile of the `Rgb` widget.

The new class `Pyramid` has been integrated, to allow processing at various scales from a single image.

Release Notes mViz 3.0.2

The minor release modifies the `Operator::Binarize` overloads and fixes a few Locator and Geometry issues (calibration).

General

The method `Status::Licenses` was always returning `true` (even though a `NoLicense` condition was raised) because of the temporary license. This behavior has been changed and it will return `false` at the first call. It will also report `false` if the license dongle has been removed.

Image Processing

Access to the pixel values in a binary image is now possible using the method Image::PixelGray1 overloads.

The prototypes of the `Operator::Binarize` methods have been redefined to include an argument `BinarizationMode` instead of the Black/White/Soft combinations. This will break existing code when these arguments were passed explicitly.

The class Operator now has a method `BiThreshold` that implements double thresholding, to separate dark, light and intermediate areas.

Template Matching

Since 2.7.2, the `Locator` was unable to find large templates because of a variable overflow. This has been fixed.

The method `Locator::Find` now supports the masked mode, allowing to define templates with don’t care areas of any shape. Check the overload taking two image arguments.

The method `Locator::Find` wasn't reporting an exact score in the `Edge` mode (in particular not exactly 1 on the training image) because of wrong handling of the pixels along the template edges. This has been fixed.

Code Reading

The `Code1DReader` supports the property `ValidCRC` that tells if the checksum was found to be valid during decoding. This is only useful for the symbologies with an optional checksum, currently `Interleaved2of5` and `Code39` (for the others it is implicitly true).

Geometric Calibration

The method `Geometry::CalibrateTarget` now automatically adjusts the scale factor for simple calibration. This means that any subsequent measurement like blob feature or gauging will be scaled accordingly. Also the `Undistort` operation will result in an undistorted image of comparable scale as the distorted source. This can break existing code where measurements are made after calibration. To cancel this behavior, call `SimpleCalibration` with no argument, or enforce your own scale factor as before.

The method `Geometry::CalibrateTarget` was failing to fit checkerboard targets in some circumstances because of a confusion in anchor coordinates computations. This has been fixed.

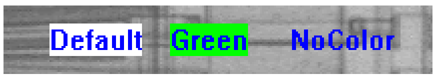
The `Geometry::Read` function will no more read the calibration files produced with earlier versions, as these are lacking sufficient calibration information.

Release Notes mViz 3.0.1

This minor release features several improvements on the `Locator` (template matching) and some graphical attributes (class `Graph` and others).

General

The attribute `Graph::FillColor` is now honored for the display of the text. The fill color, white by default, is used to fill the bounding rectangle of the displayed string. And if this property is set to `NoColor`, only the characters are shown, without an underlying rectangle.



You can now change the test size using the property `Graph::TextSize`. This sets the height of the characters, in pixels. You can also change the font by specifying a new font name with `Graph::TextFont`.

Some of the drawing functions force predefined colors by default. For instance, the Locator renders good instances in `Green` and dubious ones in `Red`. This behavior can be modified by setting the flag `Graph::AutoColor` to false.

Image Processing

Access to the individual pixel values in a binary image is now possible using the method `Image::PixelGray1` overloads.

The function `Morpho::Watershed` now has an extra overload. It allows choosing between the regions boundaries being overlaid on the source image or a pure binary edge map.

The morphological watershed has been improved for higher speed and lower stack consumption.

Template Matching

Since 2.7.2, the `Locator` was unable to find large templates because of a accumulator overflow. This has been fixed.

The method `Locator::Find` now supports the masked mode, allowing to define templates with don't care areas of any shape. Check the overload taking two image arguments.

Release Notes mViz 3.0.0

This major release provides full support for 32 and 64 bits platforms.

Compiler support

Support for the Visual Studio 2003 compiler is no more included in the SDK.

The following compilers are now supported, both for native and .NET development.

Visual Studio 2005 (32 bits)

Visual Studio 2008 (32/64 bits)

Visual Studio 2010 (32/64 bits)

Visual Studio 2012 (32/64 bits)

Should you need to use other compilers, query techsupport@visionforvision.eu.

Licensing

All licenses granted for previous version of mViz, 1.x.x and 2.x.x, will not be usable with 3.0.0 and the following.

Image Analysis

The class `Classifier` has been renamed `FeatureTable`. This class being essentially used for internal purposes, existing code should not be impacted.

The method `DrawBlob` of the class `Blobs` has a new `Label` argument that shows the blob index over the blob shape, for reference. This feature is disabled by default.

The `Region` class has new methods to create complex shapes : `Unite`, `Intersect` and `Subtract`. As their names indicates, they allow to compute Boolean combinations of existing regions. This feature combined to the Append… methods lets you create complex combinations.

The method `Region::SeedFill` now takes a `Site` argument instead of two integers for the seed point coordinates. This will break existing code.

Image Calibration

The `Geometry` class has a new method to directly process a calibration target and derive the calibration coeffcients. The target can be made of a square array of dots or a chessboard. Check `Geometry::CalibrateTarget`.

32 / 64 bits portability

mViz comes in two precompiled versions, targeting the 32 and 64 bits processors (x86 and x64 architectures), available for all supported Microsoft compilers. A 32/64 suffix appended to the filenames distinguishes them.

To avoid incompatibilities, all Sample applications are built by default for the 32 bits (x86) Platform, using the `mViz32.lib` Library or the `mVizNET32.dll` Assembly. Under the 64 bits versions of Windows, they will transparently run using the WOW64 subsystem.

For users willing to switch to the true 64 bits model, the rules below apply. Note that applications built for 64 bits will not run under 32 bits Windows (as a symptom, the error message "*An attempt to load a program with an incorrect format*." can appear.)

For native C++ development, the project files include x64 Configurations. Just select them in the Configuration Manager of Visual Studio.

For .NET applications, the "AnyCPU" model cannot be supported as there needs to be distinct Assemblies for different Platforms. The project files include x86 Configurations only. To target the x64 Platform, after creating the corresponding Configuration in the Configuration Manager, the Reference to the mViz Assembly must be changed from `mVizNET32.dll` to `mVizNET64.dll` in the project properties. The assembly mVizNET64.dll is found in the zip archive NET`\x64.zip`.

Here is a summary of the options on a 64 bits version of Windows:

Application	**Visual Studio Configuration**	**Static library**	**.NET assembly**	**Predefined Sample projects**
32 bits	x86	mViz32.lib	mVizNET32.dll	Native, .NET
64 bits	x64	mViz64.lib	mVizNET64.dll	Native
32/64 bits[1]	AnyCPU	mViz32/64.lib	mVizNET32/64.dll	-

[1] .NET applications built for the AnyCPU platform will only run if the appropriate .NET Assembly has been selected as a project Reference.