Adaptive Vision
Studio 4

intuitive ▪ powerful ▪ adaptable

software for machine vision engineers

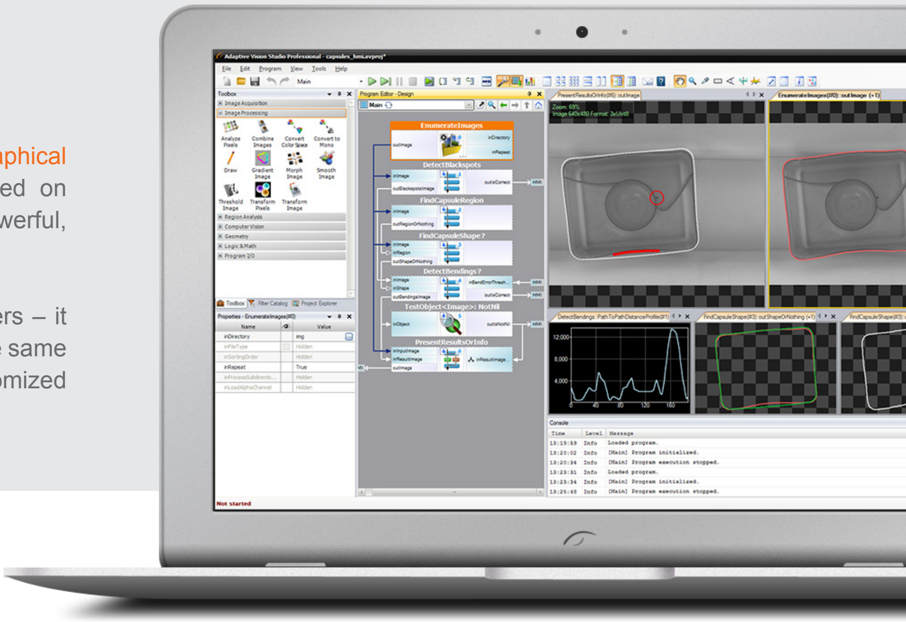
fadracer

Introduction

Adaptive Vision Studio

Adaptive Vision Studio software is the most powerful graphical environment for machine vision engineers. It is based on dataflow and comes with a comprehensive set of powerful, ready-for-use image analysis filters.

Its unique strength lies in its focus on professional users – it allows you to create typical applications easily, but at the same time makes it possible to efficiently develop highly customized and large-scale projects.



```
Time      Level  Message
13:19:59  Info   Loaded program.
13:20:02  Info   [Main] Program initialized.
13:20:34  Info   [Main] Program execution stopped.
13:23:31  Info   Loaded program.
13:23:34  Info   [Main] Program initialized.
13:25:48  Info   [Main] Program execution stopped.
```

Features

Drag and Drop

All programming is done by choosing filters and connecting them with each other. You can focus all your brain power on computer vision.

1000+ Ready-for-Use Filters

There are more than a thousand ready-for-use machine vision filters. They are organized into a well-defined category structure and comply to clear and consistent naming conventions.

Loops and Conditions

Without writing a single line of code, you can create arbitrarily complex algorithms. Loops and conditions are created implicitly with appropriate data connections.

Any-shape ROIs

Image processing operations can be performed on entire images or within regions of arbitrary shape.

Version Control Friendly

The files created by Adaptive Vision Studio have a textual format and thus allow for the use of version control systems like SVN and Git.

You Can See Everything

As the application is optimized for the needs of computer vision professionals, all intermediate results can be viewed and analysed, and parameters can be adjusted in real time.

Hardware Acceleration

The filters are aggressively optimized for the SSE technology and for multicore processors. Our implementations are ones of the fastest in the world.

Macrofilters

Large programs can be organized into small, easy to understand portions - macrofilters.

User Filters

You can use user filters to integrate your own C/C++ code with the benefits of visual programming.

Test Driven Development

A single project can contain multiple programs. The recommended strategy is to have a set of programs that test the algorithms on a set of images representing all possible variations. Assertion filters are used as the check points.

HMI Designer

You can easily create custom graphical user interfaces and thus build the entire machine vision solution using a single software package.

Comprehensive Set of Data Types

You can process not only images, but also regions, paths, geometrical primitives, profiles, histograms and more.

Modules

A collection of related macrofilters can be moved to a separate file. This constitutes a module that can be shared among many different projects.

GigE Vision and GenTL Support

Adaptive Vision Studio is a GigE Vision compliant product, supports the GenTL interface, as well as a number of vendor-specific APIs.

C++ and .NET generator

Programs created in Adaptive Vision Studio can be exported to C++ code or to .NET assemblies for easy integration with bigger software projects.

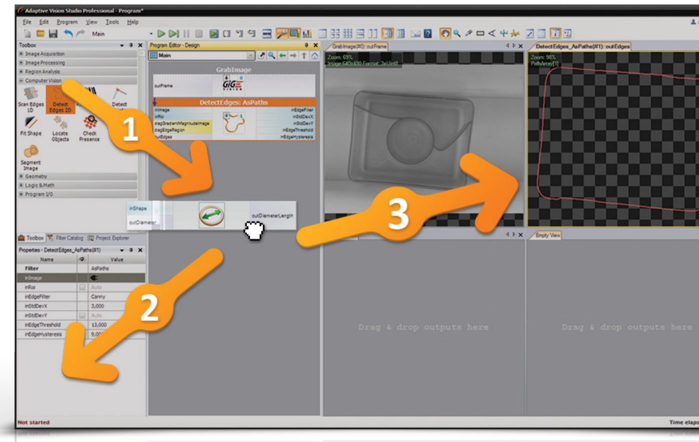
Workflow

Design the algorithm 1

Creating vision algorithms consists in repeating three intuitive steps:

1. Drag & drop filters from the toolbox to the program editor.
2. Drag & drop connections between the filters or set constant input values.
3. Drag & drop filter outputs to data preview panels for interactive analysis.

The only expertise users need to have is about the core art of computer vision – which filters to connect and how. And yet another great feature of Adaptive Vision Studio is that you can develop this competence quickly through intuitive experimentation.

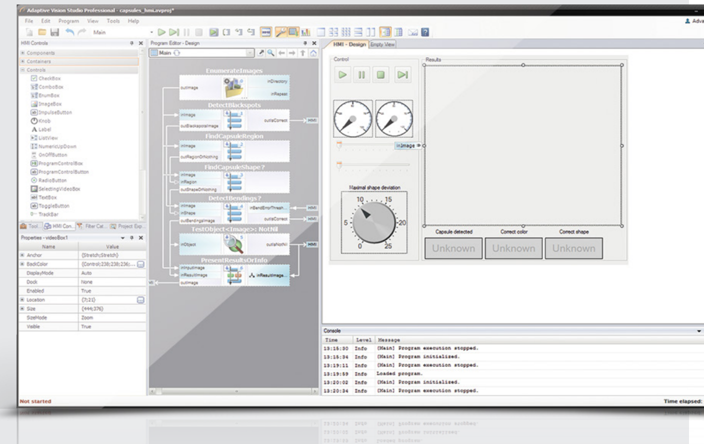


Create a custom HMI 2

Creating a custom graphical user interface is just as easy:

1. Drag & drop controls from the control catalog to the HMI panel.
2. Set the controls' properties.
3. Drag & drop connections between the filters and the controls.

Note that you can also easily integrate your algorithm with an HMI created in C++ or C#.

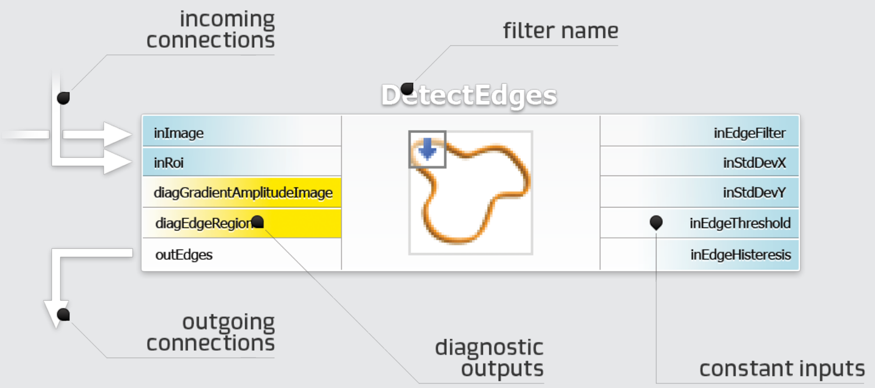


```
Console
Time      Level  Message
13:15:30  Info   (Main] Program execution stopped.
13:15:34  Info   (Main] Program initialized.
13:19:11  Info   (Main] Program execution stopped.
13:19:59  Info   Loaded program.
13:20:02  Info   (Main] Program initialized.
13:20:34  Info   (Main] Program execution stopped.
```

Deploy 3

When the program is ready, you can export it to a runtime executable and deploy it on a PC-based industrial computer or on a smart camera such as CORSIGHT from NET or NEON from Adlink.

Data Flow Programming



Introduction

Adaptive Vision Studio is a drag and drop environment designed for machine vision professionals. Experience in low-level programming is not required. Nevertheless, it is a fully fledged visual programming language and you have to understand its 4 core concepts: Data, Filters, Connections and Macrofilters.

Data & Filters

Filters are the basic data processing elements in the data flow driven programming. In a typical machine vision application input images are tranformed by a sequence of filters into regions, contours, geometrical primitives and then into final results such as a pass/fail indication. Adaptive Vision Studio also supports data collections (arrays) and filters that store information between consecutive iterations to compute aggregate results.

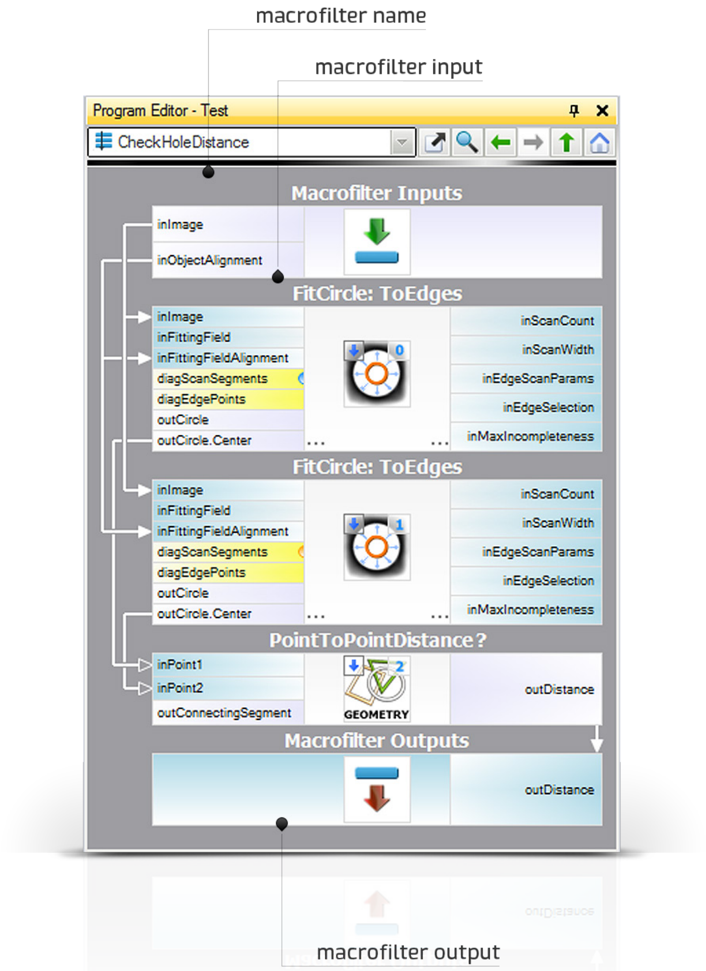
Connections

Different types of connections between filters support: *basic flow of data* →, *automatic conversions* →, *for-each processing* → and *conditional processing* →. The connection types are inferred automatically on the *do what I mean* basis.

More information can be found on our web page and in the user manual.

Macrofilters

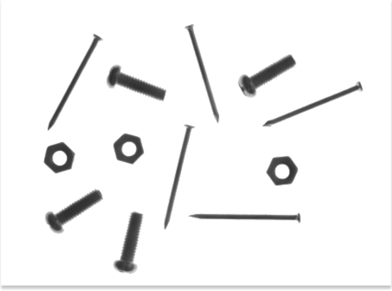
Macrofilters provide a means for building large real-life programs. They are reusable subprograms with their own inputs and outputs, and as a whole they can be used just as any regular filter. Moreover, macrofilters can perform iterative computations with state registers, thus providing an elegant way to create programs of any complexity.



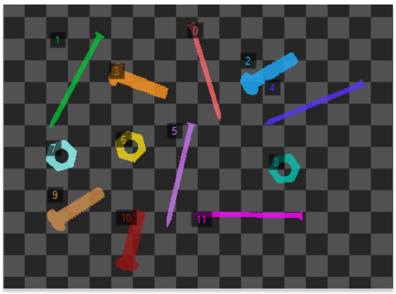
Tutorial

Here is a simple example of a data-flow driven program

1 An image is acquired from a camera:



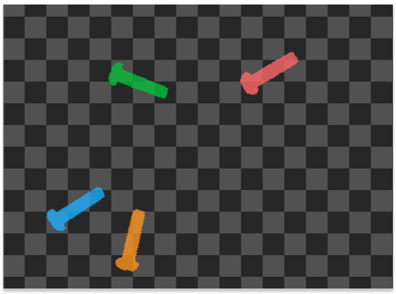
2 The image is thresholded into a region (a run-length encoded binary image) and then split into connected components:



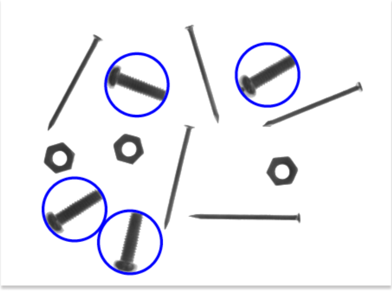
3 ClassifyRegions filter analyzes the *elongation* feature of each component:

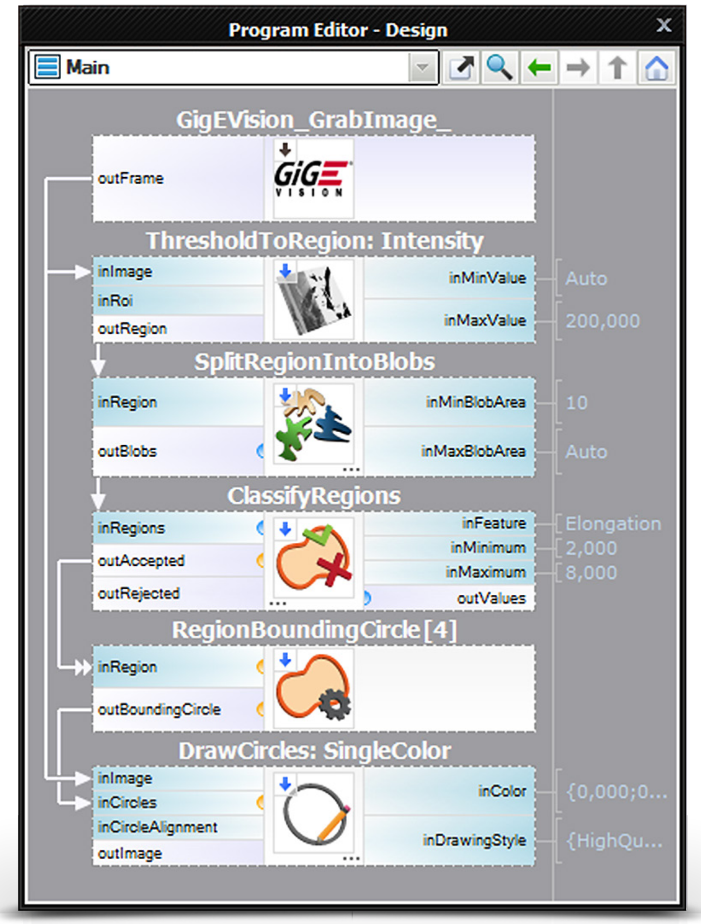
{ 19.554, 19.433, 3.866, ...}

4 ...and splits the array of regions into arrays of accepted and rejected elements accordingly to the specified range:



5 Finally the bounding circles of the accepted objects are drawn:





Program Editor - Design

Main

GigEVision_GrabImage_
outFrame

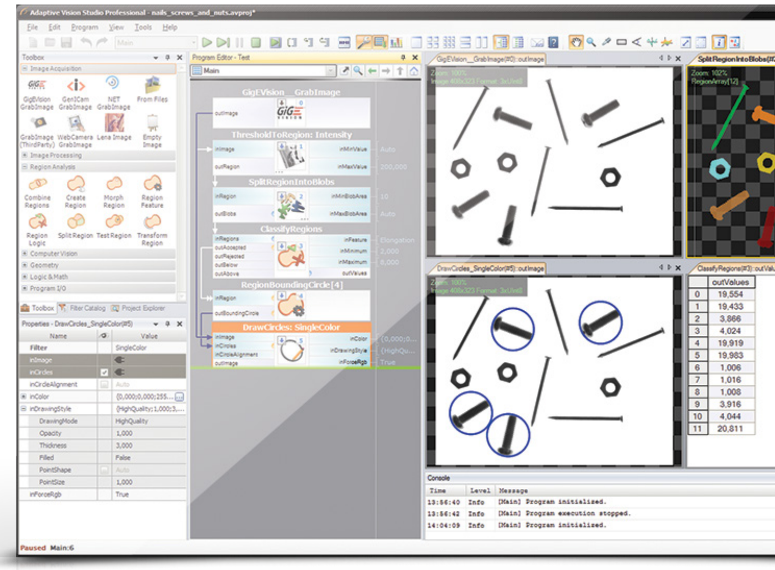
ThresholdToRegion: Intensity
inImage
inRoi
outRegion
inMinValue Auto
inMaxValue 200,000

SplitRegionIntoBlobs
inRegion
outBlobs
inMinBlobArea 10
inMaxBlobArea Auto

ClassifyRegions
inRegions
outAccepted
outRejected
inFeature Elongation
inMinimum 2,000
inMaximum 8,000
outValues

RegionBoundingCircle [4]
inRegion
outBoundingCircle

DrawCircles: SingleColor
inImage
inCircles
inCircleAlignment
outImage
inColor {0,000;0...
inDrawingStyle {HighQu...



	outValues
0	19,554
1	19,433
2	3,866
3	4,024
4	19,919
5	19,983
6	1,006
7	1,016
8	1,008
9	3,916
10	4,044
11	20,811

Console

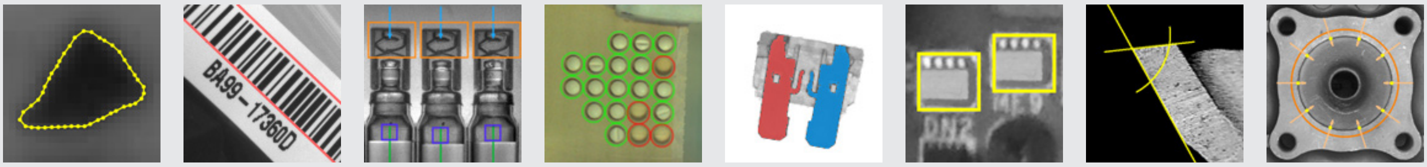
```
Time      Level  Message
13:56:40  Info   [Main] Program initialized.
13:56:42  Info   [Main] Program execution stopped.
14:04:09  Info   [Main] Program initialized.
```

Capabilities

Filters

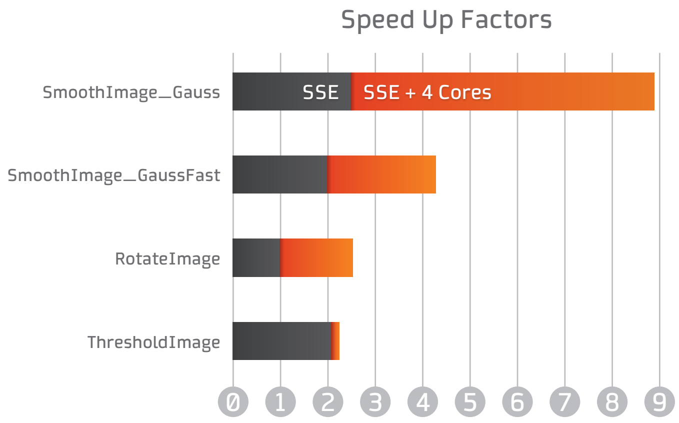
There are over 1000 filters encompassing both basic transforms and specialized machine vision tools.

- Image Processing
- Blob Analysis
- Contour Analysis
- Planar Geometry
- Shape Fitting
- Camera Calibration
- Fourier Analysis
- Hough Transform
- Barcode Reading
- Data Code Reading
- Corner Detection
- 1D Profile Analysis
- 1D Measurements
- 2D Measurements
- Gray-based Template Matching
- Edge-based Template Matching
- Histogram Analysis
- OCR
- Machine Learning
- GigE Vision and GenTL



Performance

The filters of Adaptive Vision Studio are highly optimized for modern multicore processors with SSE2 technology. Speed-up factors that can be achieved are however highly dependent on the particular operator. Simple pixel-by-pixel transforms after the SSE-based optimizations already reach memory bandwidth limits. On the other hand, more complex filters such as gauss smoothing can achieve even 10 times lower execution times than with C++ optimizations only.



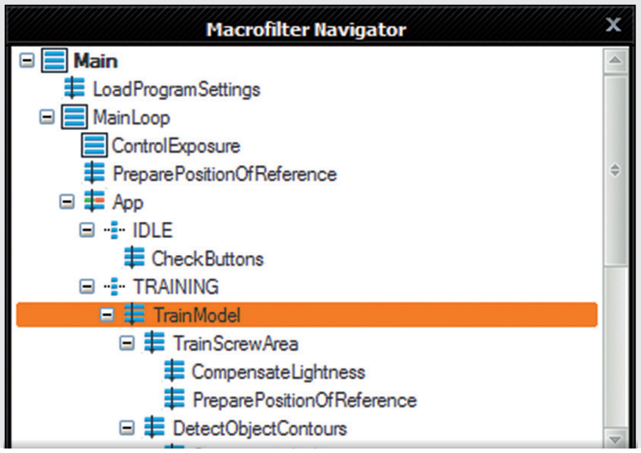
FILTER	AVS 4.0	ANOTHER PRODUCT	OPENCV 2.4.6
Image negation	0.055 ms	0.103 ms	0.055 ms
Image difference	0.050 ms	0.125 ms	0.055 ms
Rgb to Hsv conversion	0.632 ms	1.243 ms	0.586 ms
Gauss filter 3x3	0.063 ms	0.290 ms	0.439 ms
Gauss filter 5x5	0.071 ms	0.323 ms	0.661 ms
Gauss filter 21x21	0.559 ms	1.815 ms	3.132 ms
Image erosion 3x3	0.055 ms	0.127 ms	0.156 ms
Image erosion 5x5	0.059 ms	0.128 ms	0.209 ms
Threshold to region	0.061 ms	0.032 ms	
Bilinear image resize	0.205 ms	0.272 ms	0.114 ms
Splitting region into blobs	0.039 ms	0.082 ms	

The above results correspond to 640x480 resolution, 1xUINT8 on an Intel Core i7 - 3.4 GHz (4 cores) machine. To assure the most precise results big images were tested and the results were normalized. Note also that the functions from the different libraries do not always produce exactly the same output data.

More than Visual Programming

Large-Scale Projects

One of the most distinguishing features of Adaptive Vision Studio is its applicability for large-scale projects. It has been used to build algorithms composed of hundreds of tools and in systems that required maintenance of many different product versions. Creating such systems is possible due to the feature of modules, hierarchical view of the program structure, version control friendly file format and support for test-driven development.

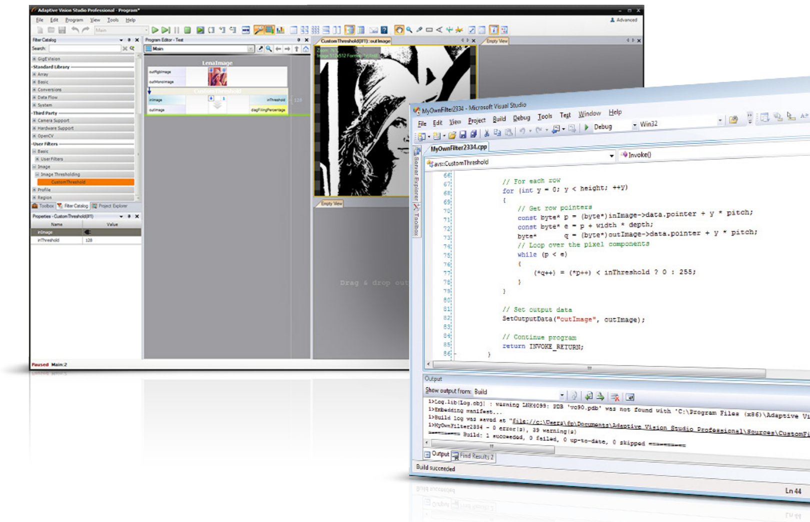


- Main
  - LoadProgramSettings
  - MainLoop
    - ControlExposure
    - PreparePositionOfReference
    - App
      - IDLE
        - CheckButtons
      - TRAINING
        - TrainModel
          - TrainScrewArea
            - CompensateLightness
            - PreparePositionOfReference
          - DetectObjectContours

User Filters

User filters allow C++ programmers to integrate their existing or new code with the benefits of the rapid graphical development. It works with Microsoft Visual C++ (including Express edition).

To add a new user filter click *File / Add New Filter* to have a sample C++ filter ready and functional in less than 30 seconds. Then on you can work side by side in both enviroments and have your filters automatically reloaded each time you build the C++ project.



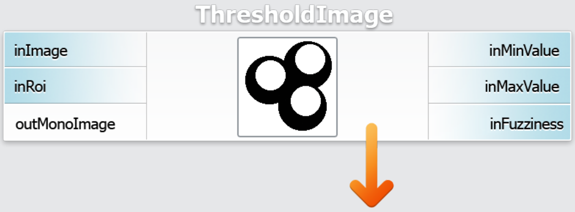
```
        // For each row
        for (int y = 0; y < height; ++y)
        {
            // Get row pointers
            const byte* p = (byte*)inImage->data.pointer + y * pitch;
            const byte* e = p + width * depth;
            byte*       q = (byte*)outImage->data.pointer + y * pitch;
            // Loop over the pixel components
            while (p < e)
            {
                (*q++) = (*p++) < inThreshold ? 0 : 255;
            }
        }

        // Set output data
        SetOutputData("outImage", outImage);

        // Continue program
        return INVOKE_RETURN;
    }
```

```
1>Log.lib(Log.obj) : warning LNK4099: PDB 'vc90.pdb' was not found with 'C:\Program Files (x86)\Adaptive Vi
1>Embedding manifest...
1>Build log was saved at "file://c:\Users\fp\Documents\Adaptive Vision Studio Professional\Sources\CustomFi
1>MyOwnFilter2334 - 0 error(s), 39 warning(s)
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```

Adaptive Vision Library

Adaptive Vision Library is a tool for C++ and .NET programmers available as a single DLL file with appropriate headers. The design is simple – for each filter of Adaptive Vision Studio there is a corresponding function in Adaptive Vision Library with several input and output parameters corresponding to the input and output ports of the filter.

The C++ code or a .NET assembly can also be generated automatically from algorithms created with Adaptive Vision Studio. This creates a unique possibility to create a program in a graphical environment and then compile it to a native Windows executable.



```
void ThresholdImage
(
        const Image&                inImage,
        Optional<const Region&>     inRoi,
        Optional<real>              inMinValue,
        Optional<real>              inMaxValue,
        real                        inFuzziness,
        Image&                      outMonoImage
);
```