

Part

1

Getting started

This part consists of the following chapters:

- ▶ [Chapter 0: About Matrox Design Assistant Help](#)
- ▶ [Chapter 1: Introduction](#)
- ▶ [Chapter 2: Building a project](#)

About Matrox Design Assistant Help

This chapter presents the Help that is distributed with Matrox Design Assistant.

- ▶ [Welcome](#)
- ▼ [Tips for using Matrox Design Assistant Help online](#)
- ▼ [Tips for using Matrox Design Assistant Help offline under Windows](#)
- ▶ [Contacting customer support](#)
- ▼ [Copyrights, acknowledgments, and patent notices](#)

Welcome

Welcome to Matrox Design Assistant Help. Its purpose is to facilitate the development of your Matrox Design Assistant project. To optimize your Help experience, see either the [online](#) or [offline](#) Help tips page.

You can access this Help at any point in the Matrox Design Assistant application using **F1** (context sensitive help). If you press **F1** over a function, reference information is displayed, otherwise User Guide information is displayed.

To familiarize yourself with the latest updates regarding this release, see the [Matrox Design Assistant Readme](#), located in the *Matrox Design Assistant* installation folder.

To quickly acquire a basic understanding of how to use Matrox Design Assistant, see:

- [Chapter 2: Building a project](#).
- The **Tutorials** on the Matrox Design Assistant **Quick Start** tab.
- The **Examples** on the Matrox Design Assistant **Quick Start** tab.
You can load each example as a project in Matrox Design Assistant and deploy it from there. To modify an existing example, save it by selecting the **Save As** option from the **File** menu and choosing another location on your hard drive.

Advanced developers requiring information regarding custom steps should click on the **Custom Step** link on the Matrox Design Assistant **Quick Start** tab.

Additional content can be found at <http://www.matrox.com/imaging>.

Matrox Vision Academy provides further video tutorials on various features of Matrox Design Assistant; these tutorials are available to registered users at https://www.matrox.com/imaging/vision_academy/da.

To view the trademark acknowledgments, see the [Copyrights, acknowledgments, and patent notices](#) section later in this chapter.



Tips for using Matrox Design Assistant Help online

This topic discusses tips on using/navigating Matrox Design Assistant Help online. You can access the Help online at https://www.matrox.com/imaging/da_help.

▼ Navigation

The following table shows the different ways to navigate Matrox Design Assistant Help:

Feature	Location	Description
Welcome page	Welcome	Gives a quick overview and provides useful links to pages in this Help and elsewhere.
Product name	In the white header	Jumps you back to the welcome page if clicked upon. Note that it indicates the Matrox Design Assistant version to which the Help applies. Whether the Help includes information for other updates is only indicated in the release notes.
Breadcrumbs	In the white header	Indicate the location and name of the current section/reference.
		Offers a hierarchical view of the Help. It has the following features:

Contents tab	On the left	<ul style="list-style-type: none"> Automatically displays your current location in the Help. Allows you to jump to a specific page in the Help. Has a context menu that allows you to collapse the current folder (Collapse) or all folders (Collapse All). To access the context menu, right-click on a contents item. If you collapse folders of the contents, you can show the location of the current topic by clicking the Locate topic context menu item.
Index tab	On the left	Allows you to enter a keyword and navigate to that topic's location(s). To navigate the available items, you can either type the word in the text box, use your mouse, or use your arrow keys.
Search tab	On the left	Allows specialized searches for locating specific information in the Help.
Collapse sections	On the right	Allows you to collapse/expand sections in Matrox Design Assistant Help. You can also use the arrow to the left of a heading to collapse/expand just that specific subsection.
Page map	On the right	<p>Offers a hierarchical view of the current page. It has the following features:</p> <ul style="list-style-type: none"> Automatically displays your current location on the page. Allows you to jump to a specific subsection on the page. Can be hidden/displayed by clicking on the Page map button in the blue bar.
Previous/next buttons	In the blue bar	Jump to the previous or next page in the Help. Click on the left or right arrow button ( ).
Tooltips	Hover over items	<p>Offer quick information. For example, you can hover over:</p> <ul style="list-style-type: none"> A title in the Contents tab, Index tab, Search tab, or Page map to display the entire title and avoid scrolling or expanding the tab. Options in the Search tab for a quick description.
Resize margins	Borders on either side of the main content.	Resize the columns on either side of any page by dragging the vertical border. To hide the Contents , Index , and Search tabs, double-click on the vertical border.

▼ Search tab

To navigate the **Search** tab:

Options	<p>Allow you to narrow the search by selecting required options. The current options are kept until you click on the Reset button.</p> <p>The Match similar words and With and without spaces options are enabled by default.</p> <ul style="list-style-type: none"> Match similar words only affects the stemming of words that are not all caps. For example, when Match similar words is enabled and Match case is disabled, TILTED will only match TILTED and tilted; however, if you search for Tilted or tilted, the search will find tilt, tilted, TILT, and TILTED. With and without spaces finds both individual and concatenated search terms. For example, blob analysis finds blob, analysis, and BlobAnalysis. <p>You can sort through the resulting list of topics according to their Title or Location by clicking on the corresponding column heading.</p> <p>To refine your query and search through previous results, enable Search previous results.</p> <p>To limit which part of the Help is searched, you can expand Search manuals and enable the required part.</p>
Popup search dialog	<p>Allows you to navigate to the results that were actually found. Things to note:</p> <ul style="list-style-type: none"> This is especially useful if you have performed a wildcard search. If there are too many hits on the page, you can search for the required word by performing an additional search with Search previous results enabled. To disable highlighting, click on the slider at the bottom-left of the dialog. You can always use your browser's search functionality to search your page.
Simple mode	<p>Enter a search term in the text box provided.</p> <p>You can hyphenate compound words; the search will find instances as a hyphenated word, 2 consecutive words, and a single word without spaces in it.</p> <p>The search supports the following operators:</p> <p>" Matches the exact phrase enclosed in double quotation marks.</p> <p>Matches 0 or more characters or symbols (for example, B*OR matches BOR and BXOR).</p> <p>* To have the search match a space, enable the Allow * wildcard to cross words option.</p> <p>Note that when you disable Match similar words and use a wildcard search (with *), you can add another * at the end of your search term to find additional characters at the end of the term. For example, the query *color will not find ColorMatcher, whereas *color* will.</p> <p>? Matches one character, space, or symbol.</p>
Regular expression mode	<p>See your browser's regular expression (regex) JavaScript syntax. You can set up searches using different groupings such as <code>\b[a-c]\w*</code>, which matches any word that begins with an a, b, or c. For more information and examples, you can visit https://www.regex101.com.</p> <p>When entering your search expression, don't write it between slashes. By default, Matrox Design Assistant Help performs global, case-insensitive searches (<code>/---/gi</code>). The regex search flags are not available; the <code>g</code> (global) and <code>i</code> (case-insensitive) options are performed internally. To match case, enable the Match case option in the Search tab (this internally removes the <code>i</code> regex flag).</p>
Max # highlighted instances per term	Sets the maximum number of results to highlight on a page per term. Decreasing this number increases the load speed of a page accessed through the search results.

Tips for using Matrox Design Assistant Help offline under Windows

The tips in this section will help you to easily print and navigate through the topics in this Help.

You can access this Help at any point in the Matrox Design Assistant application using **F1** (context sensitive help). If you press **F1** over a function, reference information is displayed, otherwise User Guide information is displayed.

If you encounter errors in Matrox Design Assistant Help, see the [Software requirements for Matrox Design Assistant Help](#) subsection of the [Minimum requirements](#) section in [Appendix B: Installation information](#).

▼ Printing

To print the Help section you are viewing:

1. Click on the **Print** button in the menu bar at the top of the Help viewer.
2. Select **Print the selected topic**.

To print an entire chapter:

1. Select the chapter name from the **Contents** tab of the Help file (table of contents).
2. Click on the **Print** button in the menu bar at the top of the Help viewer.
3. Select **Print the selected heading and all subtopics**.

To print to PDF:

1. Select what you want to print from the CHM.
2. Click on the **Print** button in the menu bar at the top of the Help viewer.
3. Select **Print the selected heading and all subtopics** or **Print the selected topic**, depending on your needs.
4. When prompted to select a printer, use the horizontal scroll bar to scroll to an appropriate PDF generating software and print to this. There are many free PDF programs to choose from if you do not already have one on your computer.

When printing multiple pages from the table of contents, and you select **Print the selected heading and all subtopics**, the Help file might stall. In this case, close and re-open the Help file.

▼ Navigating

To navigate through Matrox Design Assistant Help, use the:

- **Forward** and **Back** buttons at the top of the Help viewer to browse through topics you have already accessed.
- **Previous** and **Next** buttons in the gray menu bar at the top of the page to browse through topics according to their order in the table of contents.
- 3 tabs in the left pane of the Help viewer (**Contents**, **Search**, and **Favorites**).
- **Locate** button to display the location of the current topic in the contents tab.

The search in the Help is particularly useful when you are uncertain about where to find information about a specific topic. You can search for phrases by enclosing the phrase in quotation marks (""). You can also precisely define a search using wildcard expressions (* and ?), Boolean operators (AND, OR, NOT, and NEAR), and nested expressions (using parentheses). Note that the Help will consider a string with a period or dash as 2 consecutive words. For example, to search for X-coordinate, put it inside quotation marks to find the string.

You might find it useful to search with the **Match similar words** option enabled, and then refine your query and search through previous results. Sort through the resulting list of topics according to their title, location, or rank by clicking on the corresponding column heading.

Note that the menu item that enables the highlighting of found terms is not very intuitive. When the feature is enabled, the **Options Search Highlight Off** is available; whereas when disabled, **Options Search Highlight On** is available.

For more information on how to use the Microsoft Help Viewer, see the [Microsoft's Help on Help](#) Help file.

Contacting customer support

To address questions that are not answered in this manual, in any accompanying release note, or in release notes on any accompanying software DVD, contact your local Matrox representative or refer to the support page on the Matrox Imaging website:

<https://www.matrox.com/imaging/en/support/>.

This page provides registered customers with supplemental training resources and additional ways of obtaining support.

If your question is not addressed and you are registered, you can contact technical support. To do so, complete and submit the online Technical Support Request Form, accessible from the aforementioned page. Once the information is submitted, a Matrox support agent will contact you shortly thereafter by email or phone, depending on the problem.

If you are registered and you cannot complete the aforementioned form, you can contact Matrox customer support by e-mail at: imaging.techsupport@matrox.com.

Copyrights, acknowledgments, and patent notices

This section lists copyrights, acknowledgments, patent notices, limitation of liabilities, and disclaimers.

▼ Copyright

© Copyright Matrox Electronic Systems Ltd., 2007 - 2022.

All rights reserved.

▼ Acknowledgments

Matrox® and Matrox Design Assistant® are registered trademarks of Matrox Electronic Systems Ltd. in Canada, USA and other countries.

SureDotOCR® is a registered trademark of Matrox Electronic Systems Ltd. in Canada, USA and other countries.

Autologon for Windows® is copyright Mark Russinovich 2002 - 2011. It is produced by Sysinternals (a wholly owned subsidiary of The Microsoft Corporation).

EtherNet/IP is a trademark of ODVA, Inc.

Microsoft® and Windows® are registered trademarks of Microsoft Corporation.

Modbus is a registered trademark of MODICON, Inc.

PROFINET® is a registered trademark of PROFIBUS and PROFINET International (PI).

For the copyrights of any third-party software used or used in part by Matrox Design Assistant or its utilities, see <https://www.matrox.com/imaging/legal/third-party-software-notice-additional-terms-conditions>.

All other nationally and internationally recognized trademarks and trade names are hereby acknowledged.

Some scripts in this Help file are implemented using jQuery library versions 1.2.6 - 3.4.1 copyright © 2008 - 2019 John Resig, jQuery UI version 1.12.1 copyright © 2020, mark.js version 9.0.0 copyright © 2014–2018 Julian Kühnel, Material components version 12.0.0 The MIT License copyright © 2014-2020 Google Inc., Roboto version 2.0 Apache License copyright © 2004.

▼ Patent notice

This product might be protected by one or more patents. See <http://www.matrox.com/patents>.

▼ Liabilities and disclaimers

Limitation of Liabilities: In no event will Matrox or its suppliers be liable for any indirect, special, incidental, economic, cover or consequential damages arising out of the use of or inability to use the product, user documentation or related technical support, including without limitation, damages or costs relating to the loss of profits, business, goodwill, even if advised of the possibility of such damages. In no event will Matrox and its suppliers' liability exceed the amount paid by you, for the product.

Because some jurisdictions do not allow the exclusion or limitation of liability for consequential or incidental damages, the above limitation might not apply to you.

Disclaimer: Matrox Electronic Systems Ltd. reserves the right to make changes in specifications at any time and without notice. The information provided by this document is believed to be accurate and reliable. However, neither Matrox Electronic Systems Ltd. nor its suppliers assume any responsibility for its use; or for any infringements of patents or other rights of third parties resulting from its use. No license is granted under any patents or patent right of Matrox Electronic Systems Ltd.

Chapter 1: Introduction

This chapter focuses on setting up your development environment, and discusses the basic concepts, terminology, and documentation conventions. This chapter also provides information on how to connect to the [Matrox Design Assistant configuration](#) portal.

▼ [Introducing Matrox Design Assistant](#)

▸ [Matrox Design Assistant steps and flowchart features](#)

▼ [Licenses](#)

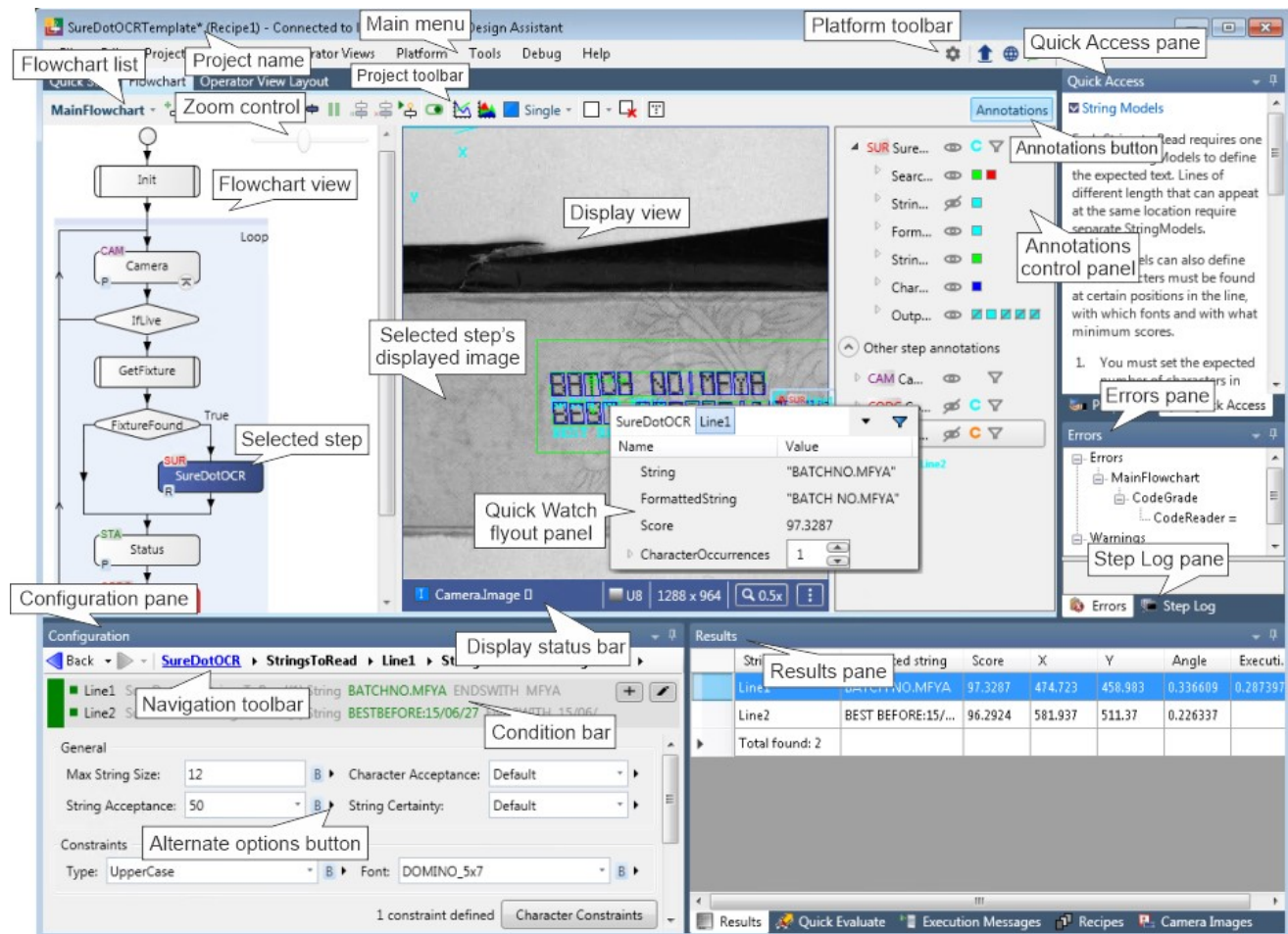
▸ [Outline of the User Guide](#)

▸ [Terminology](#)

Introducing Matrox Design Assistant

Matrox Design Assistant is a flowchart-based integrated development environment that allows you to create an imaging application without writing any code. It runs on your development computer and uses the resources of your camera and/or PC (for example, to grab an image or communicate with connected devices). The illustration of the Matrox

Design Assistant user interface below shows you the imaging application (displayed as a flowchart on the left), the image being analyzed (displayed in the middle), a list of the current annotations (displayed on the right), and a variety of interface elements typically used to design and test your application.



Using Matrox Design Assistant

With Matrox Design Assistant, you can:

- Create your project as a series of steps using a flowchart.
- Test your project from your development computer without any additional code editors or compilers and without deploying (copying and running) your project.
- Deploy your project to a Matrox Design Assistant runtime platform, such as, a supported Matrox smart camera (Matrox Iris GTR/GTX), a Matrox 4Sight EV6, or a PC connected to one or more GigE Vision or USB3 Vision cameras.
- Design and layout a web page (operator view) to receive operator input and to display your project's output.
- Run, terminate, and rerun the project on your runtime platform from within Matrox Design Assistant.

Imaging projects can:

- Grab images from your camera.
- Process images either from your camera or from disk, using several different types of processing.
- Send and receive user-defined signals from the I/O port of your runtime platform.
- Send and receive information from the serial port of your runtime platform.
- Send and receive information and save images across the network.
- Inspect a family of products without having to change projects using recipes.
- Publish results and accept user input(s) using the operator view web page.

To develop or run your project, you must connect to a runtime platform, or emulate connecting to a runtime platform. A Matrox Design Assistant runtime platform is a collection of hardware and software components on which projects can be deployed and run. For example, a supported Matrox Iris smart camera (for example, Matrox Iris GTR/GTX), a Matrox 4Sight EV6, or a PC connected to one or more GigE Vision or USB3 Vision cameras, can be considered runtime platforms. If the runtime platform has sufficient resources, more than 1 project can run simultaneously.

Design-time and runtime

Once connected to a runtime platform, your project can run in one of 2 states:

- Design-time.** At design-time, the project is running on your development computer while using the hardware services of your runtime platform; for example, grabbing images with the acquisition hardware of your supported Matrox smart camera (Matrox Iris GTR/GTX). Design-time lets you precisely control the project's execution so that testing can be concentrated within a limited section of your project. Results and messages are available within Matrox Design Assistant as the project runs.
 Note that a project running at design-time will run slower than a project running at runtime due to the relatively slow step-by-step nature of design-time.
- Runtime.** Runtime starts after the project deploys to your runtime platform. The operator view web page can be displayed in your web browser and, from there, you can both watch your project execute and interact with it. Matrox Design Assistant will continue to display runtime execution messages while the project runs on your runtime platform.

Matrox Design Assistant steps and flowchart features

Matrox Design Assistant offers a comprehensive set of steps and features to build your flowcharts.

The following is a summary of the step packages which can be added to your [flowchart](#) in Matrox Design Assistant, excluding the communication step packages:

Step category	Step	Description
Locators	CircleFinder step	Finds circle patterns (models) in the destination image using an edge-based geometric match.
	EdgeLocator step	Locates an edge in a rectangular region of the image.
	EllipseFinder step	Finds ellipse models in the destination image using an edge-based geometric match.
	Fixture step	Creates a fixture that is defined from positions, angles, and lines, or from an existing fixture with an offset.
	ModelFinder step	Finds models in the source image using an edge-based geometric match.
	PatternMatching step	Finds models in the source image using a correlation-based search.
	RectangleFinder step	Finds rectangle models in the destination image using an edge-based geometric match.
Analysis and Processing	SegmentFinder step	Finds line segment models in the destination image using an edge-based geometric match.
	BlobAnalysis step	Identifies connected regions of pixels within an image, and then calculates selected features of those regions.
	ColorMatcher step	Matches color-samples in multiple areas using different metrics in RGB, HSL, or CIELAB color spaces. The ColorMatcher step can identify, segment, or determine percentages of colors in an object.
	ImageCorrection step	Warps the input image or a region of the image to align its fixtured coordinates with the image coordinates.
	ImageProcessing step	Implements various image processing operations (such as filtering, morphology, color conversion, and color projection) to transform, enhance, and analyze images.
	IntensityChecker step	Retrieves the intensity minimum, maximum, average, and contrast in a configurable region of the image. This can be used to detect the presence of an object or markings.
	Mask step	Modify shaped regions in an image by applying a fill, smooth, or hole fill operation.
Analysis and Processing in 3D	Remap step	Converts image data to the right bit depth, and remaps from a specified input range to the full possible output range.
	AlignPlane step	Outputs modified images that are aligned to a calculated plane.
	Crop3D step	Crop regions in a point cloud or in a depth map.
	ExtractProfile step	Outputs a profile extracted from a specified line on a point cloud, depth map, or 2D intensity image.
	FillGaps step	Fill gaps (missing or invalid data points) in an image.
	Project3D step	Extracts a depth map image from a point cloud.
	Volume3D step	Calculates the volume of a point cloud's mesh or of a depth map, delimited optionally by a specified reference object.
Measurements	Measurement step	Finds edge, stripe and circle markers in an image.
	Metrology step	Fits geometric shapes (features) to regions in the image, or imports features from other steps' results. Performs measurements and tolerance checks on the features.
	BeadInspection step	Verifies the width, position, and continuity of a stream of material such as an adhesive.
Readers	CodeGrade step	Performs standard symbol (code) grading operations. It requires a link to a CodeReader step.
	CodeReader step	Reads codes from one-dimensional (1D), 2-dimensional (2D), and composite symbologies (code types). The CodeReader step handles rotated, scaled, and degraded codes.
	StringReader step	Reads text from an image.
	SureDotOCR step	Performs optical character recognition (OCR) on dot-matrix text in an image.
Classification	CNNClassIndex step	Uses a convolutional neural network (CNN) to predict the best class of an image.
	CNNClassMap step	Uses a convolutional neural network (CNN) to perform the segmentation of an image.
Image	Calibration step	Creates a calibration file from a grid in the image or from a list of points.
	Camera step	Acquires images from your 2D or 3D camera, or from disk. Optionally, associates a calibration file with the image or warps the image to correct distortions.
	CameraFocus step	Adjusts a camera's lens to a position that produces optimum focus, or only calculate the focus quality of a specified image or an image grabbed at the current lens position or manually adjust the lens position. Changing lens position requires an Iris GTR/GTX with Varioptic lens.
	CameraSettings step	Changes initial camera settings in a flowchart. This step also sets and gets feature values from GenICam compliant GigEVision, USB3Vision, GenTL and GenDC devices.
	ImageWriter step	Saves images to a shared network folder, an FTP or an unsecured HTTP server, or a folder local to your runtime platform, with an optional sidcar text file in plain text or key-value formats.
	LoadImage step	Loads images from a shared network folder, an unsecured FTP or HTTP server, or a folder local to your runtime platform.
	ToolPosition step	Specifies the tool's X- and Y-position.
	Trigger step	Generates a software trigger to grab an image.
	WhiteBalance step	Calculates the parameters needed to produce balanced grays and white.
	Break step	Breaks out of the loop, allowing your project to continue.
	Condition step	Splits the flow based on a true/false (yes/no) decision.
	Continue step	Starts a new iteration of the loop if the condition evaluates to true.
	Error step	Breaks out of the loop, going directly to the Status step.

Flow Control	Halt step	Pauses, halts, or delays the execution of the flowchart.
	Loop step	Repeats a series of steps until a specified condition is reached. Note that the Break step and Continue step can be used to control the flow of your project within a loop.
	Status step	Generates an overall pass/fail status from the specified conditions in your flowchart. When an error occurs in a preceding step, the error is caught and the status is set to fail.
	Switch step	Allows conditional flowchart execution of various sequences of steps. Path decisions are made based on the evaluation of the logical expression associated with each case.
Registration	PhotometricStereo step	Uses different lit images of a static scene to produce a single image that emphasizes substrate irregularities.
	HDR step	Merges images taken with different exposure times to create a resulting image with a greater range of detail.
Utilities	Counter step	Increments a counter in your flowchart. This step is deprecated. It is recommended to use variables as counters in your flowchart.
	DeleteFiles step	Deletes files from a list of paths.
	Navigate step	Notifies the browser displaying the operator view to switch to a different page while the project is running.
	PositionStamp step	Retrieves the value stored by one of the reference latches of an I/O command list.
	ProjectSwitcher step	Terminates the current project and switches to another project.
	Reconfigure step	This step is part of a legacy approach. For alternatives, see the Strategies for inspecting product variants section in Chapter 66: Switching products to inspect .
	ResetCounters step	Changes the value of other step inputs, including model images, color-sample images, bead inspection paths, CircleFinder step model dimensions and CodeReader step types during runtime.
	ResetCounters step	Resets the internal counters of all selected Status steps, ImageWriter steps, and Counter steps.
	Store step	Stores data to one or more variables.
	TextReader step	Allows reading text files from disk.
	TextWriter step	Allows writing text files to disk in plain text, key-value, or CSV formats. To synchronize a text file with an image file, use ImageWriter step sidecar files instead.
	TimeStamp step	Gets the current time.
Recipes and Persistence	CreateRecipe step	Creates a new recipe that is based on the recipe currently in use.
	DeleteRecipe step	Deletes the specified recipe.
	LoadRecipe step	Loads the specified recipe.
	SavePersistentData step	Saves to file the current value of all operator view input elements, variables, and reconfigured inputs that are persistent.

The following is a summary of the communication step packages that are available in Matrox Design Assistant:

Type of step	Step	Description
PLC	CommReader step	Receives data from another device (typically a PLC) using the EtherNet/IP, PROFINET, or CC-Link IE Field Basic protocol.
	CommSettings step	Modifies the online/offline state of individual communication protocols.
	CommWriter step	Sends data to another device (typically a PLC) using the EtherNet/IP, PROFINET, or CC-Link IE Field Basic protocol.
Modbus	CommSettings step	Modifies the online/offline state of individual communication protocols.
	ModbusReader step	Reads values stored in the registers used by the Modbus protocol.
	ModbusWriter step	Stores values in the registers used by the Modbus protocol.
IO	IOReader step	Receives user-defined input signals from a device connected to the auxiliary I/O pins.
	IOSettings step	Allows the modification of certain input and output signal settings during runtime.
	IOWriter step	Sends user-defined output signals to a device connected to the auxiliary I/O pins.
	TimerSettings step	Modifies a timer's settings. Hardware timers are available on certain platforms (for example, Matrox 4Sight EV6, Matrox Iris GTR/GTX smart cameras and Matrox Indio cards).
Network	NetworkConnection step	Establishes, validates or closes a TCP or UDP connection to the network. The physical network connector must be chosen in the Platform Configuration dialog.
	NetworkReader step	Reads from the network on a given connection.
	NetworkWriter step	Writes to the network on a given connection.
Robot	RobotParameters step	Retrieves parameters and position information from the last controller message received by the previous RobotWait step.
	RobotWait step	Waits for a robot controller to request a position.
	RobotWriter step	Sends a position and status to the robot controller.
	ToolPosition step	Specifies the tool's X- and Y-position.
Serial Port	SerialPortReader step	Receives data from a device connected to the serial port.
	SerialPortSetup step	Configures the serial port to be used in a subsequent SerialPortReader step and/or a SerialPortWriter step. The physical serial connector must be chosen in the Platform Configuration dialog.
	SerialPortWriter step	Sends data to a device connected to the serial port.

The Matrox Design Assistant functionality also includes:

- An Expression Editor to access hundreds of functions and operators providing math, array, string, trigonometric, logic, file, system and recipe management capabilities.
- The temporary storage of values and images in variables.
- Subflowcharts to organize and reuse sections of flowcharts and to handle events.
- Static template images loaded from files, which serve as reference parts or masks.
- **Custom** steps to implement proprietary algorithms and communication protocols. Several sample custom steps are provided (for example, the **ArrayInput** custom step, the

DateTime custom step, the **ImageRotator** custom step, and the **OCR** custom step). You can develop custom steps in the Microsoft Visual C# framework, which can call other DLLs; the DLLs need to be compiled for the runtime platform.

Licenses

There are 2 types of Matrox Design Assistant licenses: design-time licenses and runtime (MIL) licenses.

▼ Design-time license versus runtime license

A Matrox Design Assistant design-time license allows you to build flowcharts and create operator views. A design-time license also allows you to develop your projects on one development computer that is connected to the licensed platform. By default, one design-time license is available with supported Matrox smart cameras that ship with Matrox Design Assistant, and on some Matrox 4Sight units (if they come with Matrox Design Assistant installed). If you purchase a development package, it should include a Matrox Design Assistant development dongle (USB key) containing the design-time license for one development computer.

A runtime license for your runtime platform allows you to deploy and run projects on that platform. By default, a runtime license comes installed on supported Matrox smart cameras and on some Matrox 4Sight units (if they come with Matrox Design Assistant installed). The license is also available on a separate dongle or can be installed on a Matrox Concord PoE network interface board, a Matrox Rapixo CXP board, or on a Matrox Indio board.

Some Matrox Design Assistant steps are only available with certain license packages. For instance, the **ColorMatcher** step is only available when you have the Color Analysis license package. License packages allow the use of the step on both the runtime platform and the development computer. The steps available at design-time are dependent on the license packages of your current runtime platform. If you open a project that has steps that are not licensed on the active runtime platform, the project will open, but the unlicensed steps will be ignored and the project might not behave correctly. For information on how to purchase or activate a license package, see the [Managing license packages](#) subsection of this section.

Note that the Matrox Design Assistant development dongle contains all license packages. When this dongle is connected to a runtime platform, you can run a project with any Matrox Design Assistant step on that platform, and design with any step on the connected development computer.

In addition to the permanent licenses, there are 3 provisional licenses available. For more information, see the [Provisional licenses](#) subsection of this section.

▼ Managing license packages

You can only use the Matrox Design Assistant steps that your current license packages allow. To determine which license packages your current project needs, use the **Platform Project License Information** menu item when you are connected to a local or remote platform. The **Project License Information** dialog lists all step-types of your project in a tree structure. For each step-type, you can see the required license package, and the affected steps in each flowchart or subflowchart.

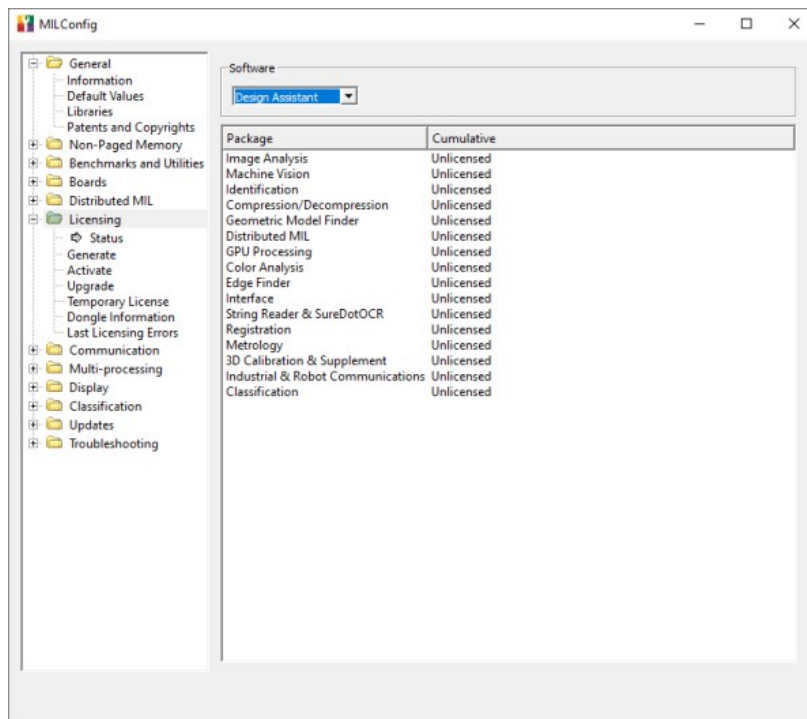
The **Project License Information** dialog also lists platform components and their required license packages; for example, it lists the Interface package for GigE Vision and the Industrial Communication package if a protocol is enabled.

The procedures to determine your current license packages or to purchase and activate new license packages is different depending on whether you have a PC platform, which includes Matrox 4Sight units, or a supported Matrox smart camera platform.

▼ With a PC runtime platform

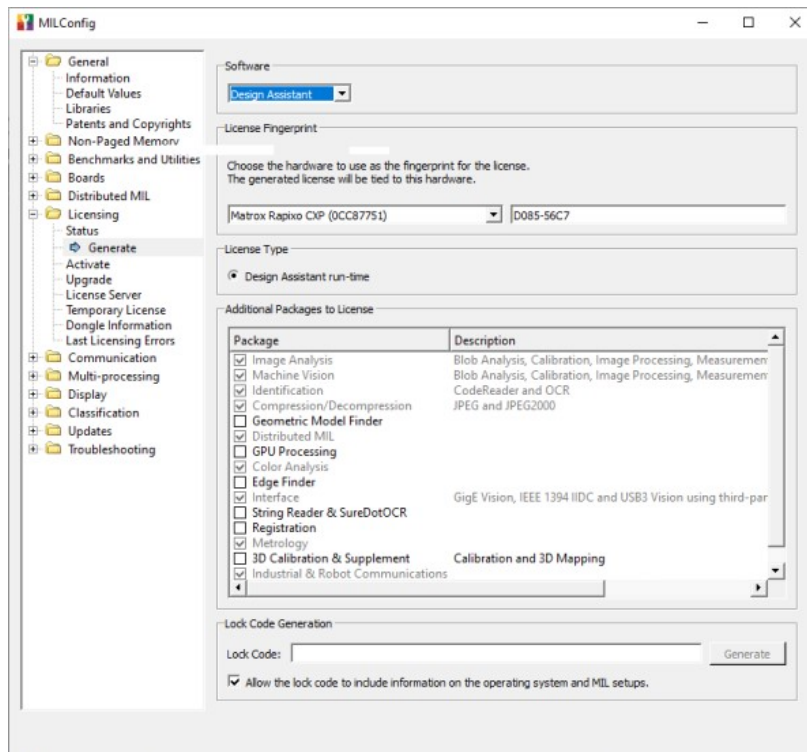
For PC runtime platforms, license information is accessible using the MILConfig utility or the **LICENSING** portal page of the Matrox Iris GTR/GTX. All available licensing options are found under the **Licensing** item in the tree structure.

To determine the license packages that you currently have, go to the **Status** page of the MILConfig utility, and select **Design Assistant** from the **Software** dropdown list. Each package has a cumulative status of either licensed or unlicensed.



To purchase a specific license package, perform the following:

1. Open the MILConfig utility from the MIL Control Center. Expand the **Licensing** item in the tree structure and then select **Generate**.
Since the MIL Image Analysis package is included in the MIL Machine Vision package, these 2 packages are mutually exclusive.
2. Select **Design Assistant** from the **Software** dropdown list.
3. Select the hardware component from which to retrieve the fingerprint, from the **License Fingerprint** dropdown list. After choosing the hardware component, select the type of license to generate and any additional packages to license.
4. Click on the **Generate** button. This will generate a lock code. Provide your local Matrox representative with the generated lock code to obtain a software license-key.
5. Activate the license. To do so, from the MILConfig utility, expand the **Licensing** item and then select **Activate**. Enter the software license-key in the text box, and then click on the **Activate** button.



Some Matrox 4Sight units come with a permanent license for most Matrox Design Assistant steps (for example, the [BlobAnalysis](#), [BeadInspection](#), [Calibration](#), [CodeReader](#), [EdgeLocator](#), [ImageProcessing](#), [IntensityChecker](#), [Metrology](#), and [PatternMatching](#) steps). Other license packages (for example, for the [ModelFinder](#) and [StringReader](#) steps) are not included. To use the corresponding steps, you can purchase the missing license package(s) through the **Upgrade** subitem.

▼ Provisional licenses

There are 3 provisional Matrox Design Assistant licenses: the 30-day design-time and runtime license, the 1-year design-time license, and the 30-day temporary license. The first 2 licenses allow the use of all steps during design-time, but only for a maximum of 24 consecutive hours at a time.

Note that changing the date and time on either a PC or a supported Matrox smart camera, while a provisional license is active, will disable the provisional license.

▼ 30-day design-time and runtime license

A 30-day design-time and runtime license-key can be obtained to run an evaluation copy of Matrox Design Assistant. This license, which is meant to allow you to evaluate Matrox Design Assistant, allows you to use all Matrox Design Assistant steps when designing or running a project on your PC. After the 30-day evaluation period, you will no longer be able to design or run projects using this license.

To request a 30-day design-time and runtime license, visit the Matrox Imaging website (<http://www.matrox.com/imaging>) or contact your Matrox representative.

▼ 1-year design-time license

The 1-year design-time license allows you to develop projects in Matrox Design Assistant without connecting to a platform. This is accomplished through emulation mode. Typically, this license is used by developers who already have a permanently licensed, supported Matrox runtime platform and want to develop new projects in emulation mode, while their camera is being used to run another project.

Note that you are not able to deploy a project to a runtime platform from Matrox Design Assistant with only a 1-year design-time license.

To enable the 1-year design-time license, enter a valid Matrox Design Assistant emulation license-key. The key is automatically obtained after registering your copy of Matrox Design Assistant using the **Support** page of the Matrox Imaging website (<http://www.matrox.com/imaging>).

Each time you start an emulation session, you will be informed of the number of days left on your 1-year design-time license. This license can be extended by renewing your Matrox Design Assistant software maintenance plan.

▼ 30-day temporary license

There is a temporary license resident on each supported Matrox smart camera and some Matrox 4Sight units; this license is valid for 30 days and cannot be renewed. When a platform has activated the 30-day temporary license, you can use all steps in both design-time and when running projects. Typically, this license is used for developers who intend to purchase a permanent license and can expect their activation key within 30 days. You should not start your temporary license until you intend to purchase an unlicensed package (for example, for the [ModelFinder](#) or [StringReader](#) step) in the next 30 days. Once the temporary license has been activated, it cannot be stopped.

To activate the temporary license on a Matrox 4Sight vision system, click on the **Start 30-day temporary license** button, found on the **Temporary License** page of the MILConfig utility. The **Status** page will then show the license packages covered by the 30-day temporary license as provisional.

To verify whether a project uses steps that are not covered by the permanent licenses, you can disable the temporary licenses using the **Suspend** checkbox. Steps that are only available with a temporary license will appear with yellow dots in the **Add Step** dialog. You must suspend the temporary license if you deploy a project on a runtime platform that will be running at the moment when the 30-day countdown expires. Failure to do so can result in an unexpected termination of the project. Suspending the temporary license does not stop the countdown.

Outline of the User Guide

After installing Matrox Design Assistant and configuring your runtime platform (a supported Matrox smart camera by itself, or Matrox 4Sight or PC connected to one or more GigE/USB3 Vision cameras), watch the introductory video tutorial available in the Matrox Design Assistant **Quick Start** tab, and read [Chapter 2: Building a project](#). It describes how to create a simple project and deploy it on your runtime platform.

The remaining chapters in the User Guide provide more detailed information about the additional features available with Matrox Design Assistant. Read these chapters as needed.

Matrox Vision Academy provides further video tutorials on various features of Matrox Design Assistant, available to registered users at https://www.matrox.com/imaging/vision_academy/da.



Terminology

Below is a list of common terms in the Matrox Design Assistant documentation. For an overview of the user interface and information about the conventions used to document it,

see the [Interface overview and documentation conventions](#) reference chapter.

- **Absolute coordinate system.** The absolute coordinate system is the default coordinate system for images in Matrox Design Assistant. The absolute coordinate system's origin (0,0) is aligned with the image's top-left corner, unless an applied calibration changes the origin. Note that, by default, results are returned with respect to the absolute coordinate system.
- **Annotation.** An annotation is a non-destructive overlay on the displayed image. Annotations are used to delineate a region on the image (for example, the search region for the [EdgeLocator](#) step), or to show results independently selected for design-time and runtime.
- **Calibration.** Calibration allows you to map pixel coordinates to real-world coordinates. This mapping can be used to get results in real-world units. The mapping can also be used to physically correct an image's distortions (see [Chapter 32: Acquisition](#)). By getting results in real-world units, you automatically compensate for any distortions in an image.
- **Compound step.** A compound step is a step that has one or more branches (for instance, a [Condition](#) step, a [Loop](#) step, or a [Switch](#) step). Any action that is performed on a compound step affects its branch step(s).
- **Debugging.** In reference to a Matrox Design Assistant project, debugging is equivalent to testing.
- **Deploy.** Deploying your project copies it to your runtime platform. It can optionally start running the project automatically, and launch your browser to display the operator view.
- **Design Assistant Agent (DA Agent).** The DA Agent is a process that manages the sharing of hardware resources, such as cameras and I/O ports, between the design-time and the runtime.
- **Design-time.** Design-time is the time in which you can create your flowchart and test its steps, individually or in sequence, before deploying the project on your target platform.
- **Development Computer.** A development computer is a computer with the Matrox Design Assistant Integrated Development Environment installed, being used to create projects for deployment on a runtime platform.
- **Element.** Elements are items used to create your operator view. There are output elements, input elements, and static elements. An output element is a value, a PassFail marker, or an image display item. An input element is a button, a radio button, or a text box. A static element is an image.
- **Emulation mode.** In Emulation mode, you can work with the Matrox Design Assistant application without connecting to a runtime platform. You can build and configure your flowcharts, and test processing steps. However, Emulation mode only allows you to work with stored images, and all the communication steps of your flowchart will be skipped.
- **Execution marker.** When the project is running, the execution marker is an orange or red outline that appears around a step that is currently being executed. When the project is not running, the execution marker appears around the step that was last executed.
- **Expression.** An expression specifies a computation using a sequence of numbers, strings, booleans, constants, objects, or logical operators. Once evaluated, the expression returns the computed result. This can include a simple link to a variable in your project, an input or output of any step of your project, or a platform configuration setting. The [Advanced](#) editor of the [Configuration](#) pane displays lists of operators and functions from which you can build an expression.
- **Fixture.** A fixture is a reference frame consisting of an (x,y) position and an X-axis direction. When a fixture is associated with a step, all position and orientation inputs of the step are defined with respect to the fixture. Fixtures are used to allow a step's regions to be tied to an object in the image, so that when the object moves or rotates, the regions will follow accordingly.
- **Flowchart.** A flowchart is a diagram that identifies the starting and ending points of a project, the sequence of steps in the project, and the decision and branching points along the way. The main flowchart is created with a loop, a [Camera](#) step, and a [Status](#) step. Subflowcharts are created with flowchart start and flowchart end nodes.
- **Inspection site.** The inspection site is the set of Matrox Design Assistant project parameters associated with a project deployed to a particular runtime platform instance. For example, 2 supported Matrox smart cameras or 2 Matrox 4Sight EV6s on 2 separate work cells.
- **Internet Information Services (IIS).** Used by Matrox Design Assistant 4.0 and later, the IIS publishes the portal pages and operator view pages on a PC runtime platform.
- **Link.** A link is an expression that references an input, output, variable, or platform configuration setting. Each time an input containing a link is evaluated, it can give a different result depending on the current value of the link. A simple link is composed of identifiers separated by a period.
- **Matrox Design Assistant configuration portal.** The [Matrox Design Assistant configuration](#) portal is a set of web pages, referred to as portal pages, resident on the runtime platform. The portal allows you to manage projects deployed to your runtime platform, view images from connected cameras, view industrial communication information (such as auxiliary I/O signals), define calibrations, and manage project security. A supported Matrox smart camera's configuration portal has additional administrative and configuration options.
- **MILConfig utility.** The MILConfig utility is a MIL utility which allows a Matrox Design Assistant user to specify default cameras, verify licenses, fetch software updates, and generate troubleshooting logs.
- **NIC.** A NIC is a network interface controller. Some NICs support settings needed for efficient data transfer from GigE Vision cameras. Add-on cards (adapters) can have between 1 and 4 NICs (for example, Matrox Indio and Matrox Concord PoE), while industrial PCs can have more (for example, Matrox 4-Sight GPM has 6).
- **Operator view.** The operator view is your deployed project's interface. The operator view is used by the operator (user) to view results and interact with the running project. You can have more than one operator view per project. When you deploy your project, the operator views are copied to your runtime platform as web pages.
- **Persistence.** A persistent value retains its value (does not reset) every time the project is run. The term persistence can sometimes be referred to as static (such as static variables).
- **PLC.** A PLC is a programmable logic controller. It is typically used to manage manufacturing processes, and is usually programmed with ladder style logic, providing sequential and combinatorial control of devices through discrete I/O control, or via Ethernet based industrial communication protocols.
- **Portal Pages.** The web pages that make up the [Matrox Design Assistant configuration](#) portal.
- **Preview.** Whenever a step's property is changed, Matrox Design Assistant will use the new value to generate a preview of the image and its annotations. The contents of the [Results](#) pane are not updated during preview, only when the step is run or rerun.
- **Project.** A project stores all the relevant information, such as the flowchart, operator views and MIL contexts, that relate to an application that will run on your runtime platform.
- **Quick Access.** Quick Access is your primary guide for configuring each step in your project. Quick Access links to different locations within Matrox Design Assistant, taking you directly where you need to go.
- **Recipe.** A recipe is a group of step inputs used to inspect one variant of a family of related products. Each recipe has a unique ID and a unique name.
- **Runtime.** Runtime is the time after a project is deployed on your runtime platform. An operator can interact with your running project's operator view. You can start or stop runtime projects from the [HOME](#) portal page, or a connected development computer.
- **Runtime environment.** The runtime environment is the support software running on the runtime platform, including DA Agent and a web page server.
- **Runtime platform.** A Matrox Design Assistant runtime platform is a collection of hardware and software components on which projects can be deployed and run. For example, a supported Matrox Iris smart camera (for example, Matrox Iris GTR/GTX), a Matrox 4Sight EV6, or a PC connected to one or more GigE Vision or USB3 Vision cameras, can be considered runtime platforms. If the runtime platform has sufficient resources, more than 1 project can run simultaneously.
- **Step.** A step is a Matrox Design Assistant building block composed of one or more properties. Step configuration is the process by which a step's properties are set. The current (or last) step that was run is outlined in orange in your flowchart. This outline is known as the execution marker. Note that when an error occurs, the execution marker is red. The execution marker moves as each step is executed. Clicking on a step selects it. A selected step is dark blue in color. When you select a step, you can configure its properties.

Chapter 2: Building a project

This chapter presents an overview of how to build an application with Matrox Design Assistant.

- [Building a project overview](#)
- ▼ [Connecting to a runtime platform](#)
- ▼ [Creating, opening, or importing a project](#)
- ▼ [Cameras and your runtime platform](#)
- ▼ [Building the flowchart](#)
- ▼ [Techniques for running the flowchart and accessing results](#)
- ▼ [Cameras and images](#)
- ▼ [Display features during design-time](#)
- ▼ [Real-world units and fixtures](#)
- ▼ [Search regions](#)
- ▼ [Dealing with color images](#)
- [Status step](#)
- ▼ [Events and actions](#)
- ▼ [Specifying reset points for variables and step outputs](#)
- ▼ [Inspection labels](#)
- [Communication](#)
- ▼ [Testing and debugging a project at design-time](#)
- [Setting up the operator view](#)
- ▼ [Deploying and running your project](#)
- ▼ [Using the Matrox Design Assistant configuration portal](#)

Building a project overview

A Matrox Design Assistant project typically acquires an image or 3D data, processes it, makes a decision, communicates with external devices using the I/O, serial or networking ports, and reports the results to a web page (the operator view) on your runtime platform. These are the basic steps for building such a project:

1. [Connect to a runtime platform](#) (from within Matrox Design Assistant) or start in emulation mode.
2. [Create a calibration file](#), if using real-world measurements, from the **DEFINE CALIBRATION** tab of the **TOOLS** portal page of the **Matrox Design Assistant configuration** portal.
3. [Create, open, or import a project](#).
4. [Configure the Camera step](#) and verify the focus and lighting for your camera while [acquiring live images](#). You can also configure the **Camera** step to use test images from disk ([image sets](#)). Remember to select your calibration. At runtime, you typically change the **Camera Source** to acquire images from the camera.
5. [Build the flowchart](#), which consists of one or more steps that can include processing, analysis, communication with external devices, and/or control of your camera.
6. Use the **Quick Access** pane to access instructions related to setting up the selected step and its associated properties. You can also use it to access specific properties of the selected step and configure them in the **Configuration** pane.
7. [Use the displayed image\(s\)](#) to specify [search regions](#) for a specific step (for example, a search region for an **EdgeLocator** step, or a test region for an **IntensityChecker** step).
8. [Test your project](#).
9. [Update the operator view](#) to match the input and output requirements of your project.
10. [Deploy and run your project](#) to watch the operator view in action.
11. Use the **Matrox Design Assistant configuration** portal web pages to manage all the projects that have been deployed to the platform. It is also used to monitor the internal state, communication, and events of the runtime platform.

Save your project each time you make significant changes. Projects are automatically saved just prior to deployment. An asterisk (*) next to the project name in the title bar indicates that there are unsaved changes. When saving, wait for the asterisk to disappear before shutting down the computer, or files can be corrupted.

It is highly recommended to review the examples and template projects that relate to your project. Examples mostly focus on one feature working with saved images, although some examples also show how to work with images grabbed from the camera. Examples on working with 3D data (point clouds and depth maps) are also available.

Once you are familiar with Matrox Design Assistant projects, the template projects can prove an invaluable resource for developing complete applications, including PLC communication, user-defined models for fixturing, step configuration, and many other techniques. For presence/absence, code reading, and length/width measurement, you might be able to use a template project directly after some cosmetic changes. For more information, see the [Using the existing Matrox Design Assistant template projects](#) subsection of the [Creating, opening, or importing a project](#) section later in this chapter.

Connecting to a runtime platform

Before working on a project ([design-time](#)), you must connect to a runtime platform, or emulate connecting to a runtime platform. A Matrox Design Assistant runtime platform is a collection of hardware and software components on which projects can be deployed and run. Some components are required and some are optional.

Note that if the runtime platform has sufficient resources, more than 1 project can run simultaneously. For more information, see [Chapter 72: Running multiple projects on a runtime platform](#).




▼ Runtime platform components

Matrox Design Assistant can use a PC or a supported Matrox Iris smart camera as a runtime platform (for example, Matrox 4Sight EV6 and Matrox Iris GTR/GTX, respectively). When using a PC as the runtime platform, it should be connected to one or more GigE Vision cameras, USB3 Vision cameras, or other supported 3D sensors. In the latter case, if your sensor is not directly supported, contact your Matrox sales representative. A runtime platform must also have a network component. In addition, Matrox Design Assistant can make use of some optional hardware components, if present, such as digital I/Os, serial ports, and accelerators for industrial communication protocols (for example, the Matrox PROFINET Engine).

Some hardware components (such as [Matrox Indio](#)) must be represented in a Matrox Design Assistant project with a system component before you can access the physical component (for example, any auxiliary input/output signals). Typically, this happens automatically.

A runtime platform must also include a few software components, which are referred to collectively as the runtime environment. The most important parts of the runtime environment are:

- **DA Agent.** A continuously running service that manages communication and hardware access to the runtime platform. It is also responsible for starting and stopping deployed projects and project change validation. It acts on behalf of a running Matrox Design Assistant design-time IDE. The status is shown on a PC runtime platform by the color of the Matrox Design Assistant notification icon in the taskbar. The icon has 3 colors.

Color	Icon	Description
Red		The task has stopped.
Gray		There are licensing issues.
Green		The task is running.

- Website serving the [Matrox Design Assistant configuration](#) portal pages and the running project's operator view pages. If the website cannot start (for example, if some other process has taken the http port 80), then a warning triangle appears over the DA Agent icon:

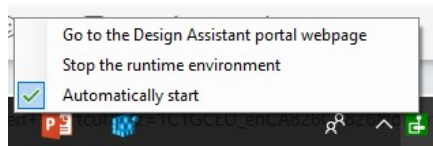


Matrox Design Assistant setup allows you to install the design-time environment, the runtime environment, or both, on a computer.

At design-time, Matrox Design Assistant runs on the development computer and communicates with the runtime platform, through the DA Agent, to access its hardware components.

▼ DA Agent context menu

You can right-click on the Matrox Design Assistant notification icon in the taskbar to open the **DA Agent** context menu.



From here, you can quickly go to the [Matrox Design Assistant configuration](#) portal, stop the runtime environment, or specify whether or not to automatically start the runtime environment.

▼ How to connect to a runtime platform

When you start Matrox Design Assistant, you have the option to connect, or emulate connecting, to a runtime platform, and you will be prompted to connect to a runtime platform when you open, create, or import a project. You can connect by selecting one of the following options in the prompt or in the **Platform** menu:

- **Connect local.** Establish a connection between the design-time and the runtime environment installed on your computer. This connection allows Matrox Design Assistant to communicate with one or more GigE/USB3 Vision cameras (such as Matrox GatorEye). Connecting to your local computer allows you to design and test your project locally, deploy on your local computer, and use any connected GigE/USB3 Vision cameras to run your project.
- **Connect remote.** Establish a connection between the design-time environment and runtime environment on a remote runtime platform (either a supported Matrox Iris smart camera or a remote computer). This connection allows Matrox Design Assistant to communicate with the remote runtime platform, allowing you to test and eventually run your project (deploy it).
- **Start emulation.** Run Matrox Design Assistant in emulation mode. This allows you to run Matrox Design Assistant without connecting to a runtime platform. You can build and configure flowcharts and test processing steps, using images from files.

▼ Requirements to connect

Upon connecting, Matrox Design Assistant will look for the appropriate runtime [license](#). When connecting locally or emulating a connection, if no valid license is found, you will be prompted to manually enter a valid license-key. When connecting to a remote computer, it must have the appropriate runtime licenses; otherwise, you will not be able to connect. To establish a connection with the local PC or a remote PC (runtime platform), the Matrox Design Assistant notification icon in the Microsoft Windows taskbar of the PC runtime platform must be green (as previously indicated).

▼ Switching runtime platforms once connected to a runtime platform

Once connected to a runtime platform, you can switch between runtime platforms using the **Platform** menu. If a project is open when you switch runtime platforms, the project will ask you to save any recent changes, and then it will close itself before the runtime platform is switched. Once you are connected to the new runtime platform, the project will open again.

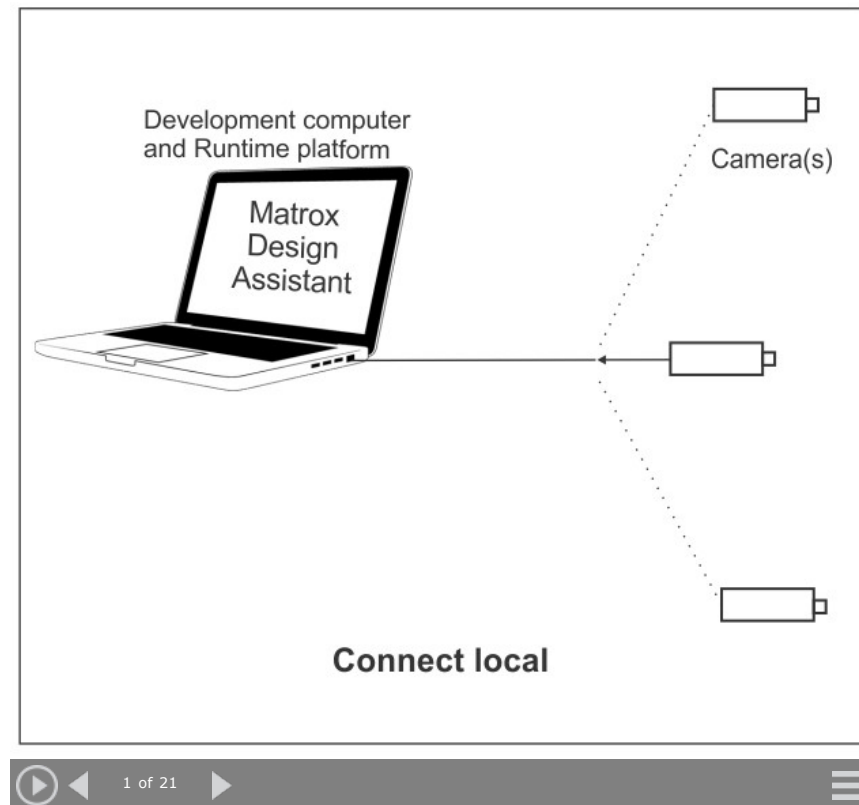
▼ Connecting with your local PC as your runtime platform

To use your local computer as your runtime platform, click on **Connect local** in the **Platform** menu. This uses your local computer as the development computer and sets up the conditions to allow you to deploy your project locally.

If an error occurs when you try to connect to your local computer, refer to the [Appendix C: Troubleshooting](#) appendix or the Troubleshooting section of your [Matrox Design Assistant Readme](#) file.

Once Matrox Design Assistant (through the DA Agent) establishes a connection with the runtime environment on your local computer, you can click on the **New Project**, **Open Project**, and **Import Project** links in the **Quick Start** tab, and *Connected to localhost* appears in the title bar of the window.

The following animation demonstrates using a local computer as a runtime platform.



▼ Connecting with a remote PC or Matrox smart camera as your runtime platform

To use a remote computer as a runtime platform, use the **Platform Connect remote** menu item. This will open the **Connect to platform** dialog. To create the connection, you must specify the network name or the IP address of the remote PC or Matrox smart camera.

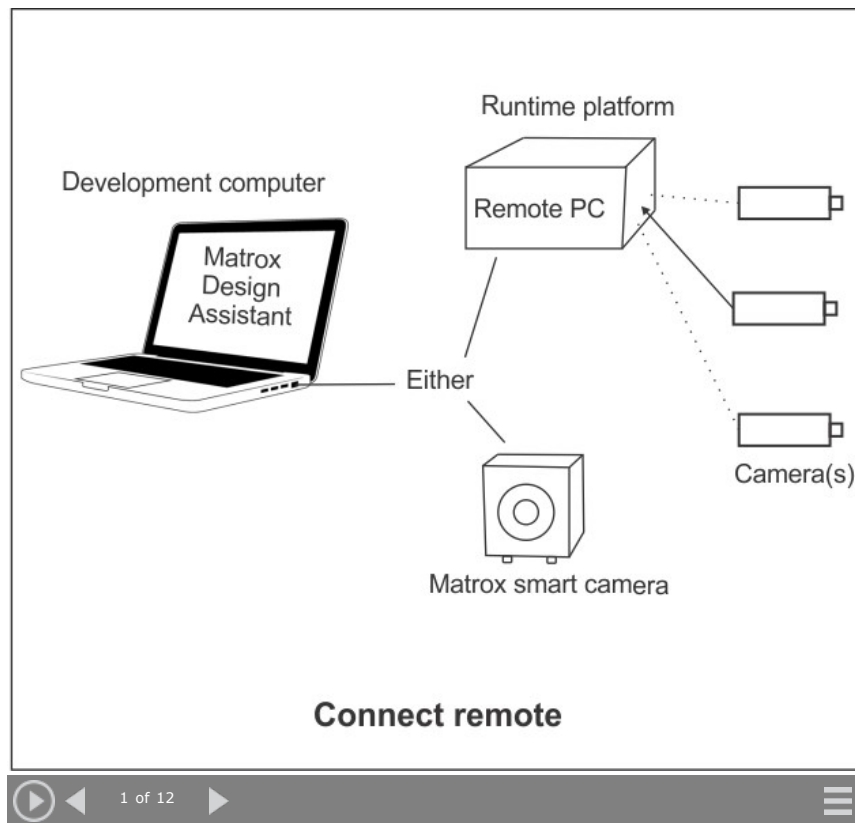
Note that to be able to connect, the remote PC must have an installed runtime environment of the same Matrox Design Assistant version and a build number that is the same or lower.

If an error occurs when you try to connect to a remote computer, refer to the [Appendix C: Troubleshooting](#) appendix or the Troubleshooting section of your [Matrox Design Assistant Readme](#) file. If an error occurs when connecting to a Matrox Iris GTR/GTX smart camera, refer to the Troubleshooting section of the [Getting Started with Matrox IrisGTR](#) readme file.

Once Matrox Design Assistant (through the DA Agent) establishes a connection with the runtime environment on your remote computer, you can click on the **New Project** and **Open Project** links on the **Quick Start** tab and *Connected to PlatformName* appears in the title bar of the window.

At runtime, a project can access the name of the runtime platform on which it is running, using the `HOSTNAME` function.

The following animation demonstrates using a remote computer as a runtime platform.




▼ Emulation mode

In emulation mode, you can work with Matrox Design Assistant without connecting to an actual runtime platform; however, in this mode, you can only work with stored images, and all the communication steps will be skipped.

Although you do not connect to an actual runtime platform, emulation mode lets you mimic one, such as a supported Matrox smart camera or a Matrox 4Sight EV6, using the [Platform Configuration](#) dialog. By specifying a particular runtime platform type, it is possible to configure hardware-specific features such as the auxiliary I/O Engine. This can be useful when working on a project that has I/O pins in the design. By emulating the runtime platform for which the project was originally designed, you can ensure that your project will not generate errors due to an insufficient number of I/O pins (allowing you to work on, or debug, other aspects of such projects).

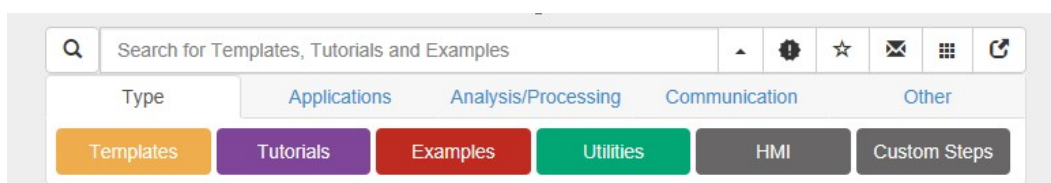
To use emulation mode, click on **Start emulation** the **Platform** menu. Once emulation mode begins, you can click on **New Project** and **Open Project** links on the **Quick Start** tab, and "Emulation started" appears in the Matrox Design Assistant title bar. Although Matrox Design Assistant is fully functional in emulation mode, note that:

- You cannot grab live images. Instead, use [image sets](#) to select any image(s) to work on.
- You cannot access communication hardware; **IO** steps, **Serial Port** steps, **Network** steps, **PLC** steps, the [TextReader](#) step, the [TextWriter](#) step, and the [ImageWriter](#) step will be skipped. This is equivalent to setting them offline using the [Communications](#) () toolbar button in the **Platform** toolbar.
- An emulation session lasts for 24 hours at a time. If a project is opened after that period, it will be automatically saved and closed, and emulation mode will end.
- You can end emulation mode using the **Platform End Emulation** menu item, which is available when there is no open project.

Creating, opening, or importing a project

There are 3 ways you can open projects with Matrox Design Assistant. You can create a new project on your development computer, you can open a project that has been previously saved, or you can import a project from your runtime platform. A project is copied to your runtime platform when you [deploy the project](#), so importing a project includes the most recent changes that have been made to the deployed project.

To create, open, or import a project, use the **New Project**, **Open Project**, or **Import Project** menu item in the **Quick Start** tab, or open a recent project under **Recent Projects** after connecting to a runtime platform. To access sample projects, click on the **Examples** button in the **Type** tab of the **Quick Start** tab.



If your development computer has internet access, you can view new and updated sample projects, templates, and tutorials in the **Quick Start** tab. Depending on the content, you will be able to use the **Download and open project** button or the **Upgrade example project** button to access this new content. To view only new and updated content, click on the **Only show me new and updated topics** (🔔) button at the top-right of the **Quick Start** tab.

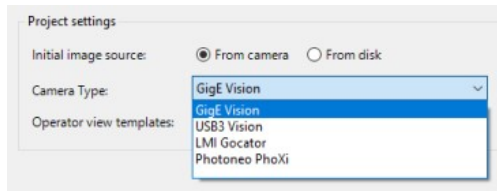
To search for examples, templates, or tutorials that use a specific step, use the search bar. You can also search for examples, templates, and tutorials by their name.

You can also access the options of the **Quick Start** tab in a separate window. This is useful if you are watching a video tutorial and need to follow along in Matrox Design Assistant. To access the options of the **Quick Start** tab in a separate window, click on the following **Launch** (🔗) button, found at the top-right of the **Quick Start** tab:

▼ Details about creating and opening a project

To create a project, follow the steps below using the **New Project** dialog, accessible from the **Quick Start** tab.

1. Enter a **Name** and **Location** for the new project. Use the **Browse...** button to choose a location on your development computer.
2. Specify **Project settings**.
 - Choose an **Initial image source**. Specify whether your project will run using images from a camera (**From camera**) or images stored on disk (**From disk**). Selecting **From disk** sets your **Camera** step to use **image sets**. Note that you can switch from an image set to a physical camera source (or vice-versa) once working on the project.
 - Choose a **Camera Type**. Note that you can select a different camera once working on the project.



The content of the **Camera Type** list can vary depending on which 3D sensors are installed on your computer.

- Choose an **Operator view template**. For information on **Operator view templates**, see the [Operator view templates and the default view](#) subsection of the [Page structure](#) section in [Chapter 63: Customizing the operator view](#).
3. Click **OK**. At this point, the interface refreshes and the **Flowchart** tab displays the default flowchart containing a **Camera** step and a **Status** step within a loop.

You can open an existing project from the **Open** dialog, accessible from the **Open Project** menu item in the **Quick Start** tab.

When opening a project with steps that are not permanently [licensed](#), you are notified of the time left in your provisional license. When the provisional license expires, you can open projects with the now unlicensed steps, but they will be disabled, and you are notified about the package(s) that must first be licensed.

▼ Details about importing a project

If you do not have a copy of an existing project at design-time, you can import a previously deployed project. When you import the project, it will also import its [validation set](#) if it exists. To import a previously deployed project from your runtime platform to your development computer, select it from the **Import a project** dialog, accessible from the **Quick Start** tab by selecting **Import Project**, or from the **File Import Project** menu item, and provide the location to store it on your development computer. When importing a project, it automatically opens. You must be connected to the platform before importing a project from the platform.

A deployed runtime project that is being used or modified can contain different data from the project at design-time. While a deployed project is running, it is accumulating its own data stores. Importing a project from a runtime platform will import the most recent version of the project and any [Project Change Validator](#) sets, including any changes made to the project, its recipes, and its settings. **File Import Project** is used to get a design-time copy of a project you don't already have. If you want to get the latest persistent settings and recipes from a platform for a project you already have, do not use **File Import Project**. Instead use **Recipe Synchronize** and **Recipe Import**. For more information about synchronizing and importing recipes from your runtime platform, see the [Recipes](#) pane.

To import a project from a previous (major) version of Matrox Design Assistant, you must install that version (side-by-side with the newer version), import the project, and then upgrade it using the newer version.

▼ Loading projects from earlier versions of Matrox Design Assistant

Loading a project from a previous version of Matrox Design Assistant will prompt you to upgrade. Once a project is upgraded, it cannot be loaded into a previous version of Matrox Design Assistant. Keep a backup copy of the original project to continue supporting installed cameras or systems that cannot be upgraded (due to company policies or procedures). For more information, see the [Matrox Design Assistant Readme](#).

▼ Using the existing Matrox Design Assistant template projects

Matrox Design Assistant comes with several template projects (accessible from the **Quick Start** tab) that implement some complete inspection applications (for example, **CodeReader**, **Presence Absence**, **Measurement**, **SureDotOCR**, and **ColorMatcher**). Template projects are ready to run, and can serve as a starting point for your own applications. With a template project, you can use your own setup, lights, and images to test and implement your project.

▼ Common features shared by all template projects

While each template project provides specific functionalities, they all share the following features accessible from the operator view:

- Perform a live capture with a camera, and change its exposure time.

- Grab images from the live capture and save them to your runtime images folder.
- Change your image source between saved images and physical camera.
- Create a fixture using either the **PatternMatching** step or **ModelFinder** step.
- Train models for the **PatternMatching** step or **ModelFinder** step. You can interactively define models so that these steps output an appropriate fixture.
- Train regions. You can interactively create search regions or regions of interest (ROIs), optionally fixtured to model occurrences.
- Create recipes at runtime, as well as change the current active recipe and manage its settings.
- Manage communication with a PLC (online/offline).
- Show how to set application specific pass/fail conditions.

These features cover the basic needs of most imaging applications. For more information on using them, see Matrox Vision Academy, available to registered users at https://www.matrox.com/imaging/vision_academy/da.

▼ Using template projects to perform a live capture with a camera

Before deploying a template project, you should verify that its system and camera platform settings match your hardware (for example, Matrox Iris GTR/GTX, GigE Vision, USB3 Vision, or Matrox Rapixo CXP). For more information, see the **Platform settings for your camera** section in **Chapter 32: Acquisition**.

▼ Using template projects with images

Template project image sets are configured by default to use one subfolder per recipe, with the same name as the recipe name images folder that is in the templates project folder. To use your own images with a template project, you must copy the image files to the proper location (folder) for that project. When doing so:

- Choose the name of the recipe (folder) you will create; for example, if you are adding images to the **CodeReader** template project, you might use *DemoCodes*. You can also reuse one of the existing recipe names.
- Copy your images to the chosen folder. For example, to add your demo images to the **CodeReader** template project, you would copy them to `C:\Users\Public\Documents\Matrox Imaging\DA 7.0\Example Projects\CodeReaderTemplate\Images\DemoCodes`.

For more information about using images (instead of acquiring images from a camera), see **Chapter 33: Acquisition from image sets**.

Cameras and your runtime platform

To set up the hardware and software components of your runtime platform for use within your project, use the **Platform Configuration** dialog.

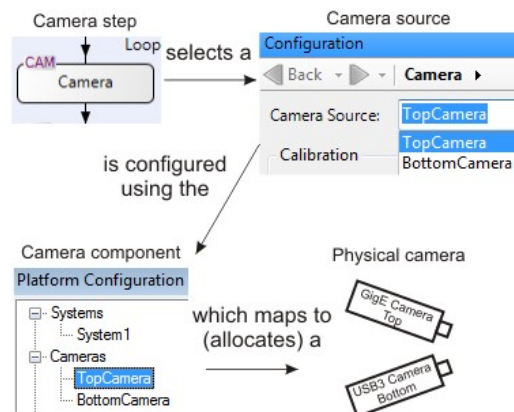
This dialog is accessible by clicking on the **Project Configure platform** menu item, or the **Configure platform** (⚙️) toolbar button in the **Platform** toolbar at the **top-right of the interface**.

Note that, if you want to simultaneously run multiple projects on a runtime platform, and more than one of those projects grab from a camera, you must explicitly set, for each project, the camera allocation mode to a specific (and different) camera. For more information, see the **Considerations when running multiple projects simultaneously** section in **Chapter 72: Running multiple projects on a runtime platform**.

▼ Allocating cameras on a PC runtime platform

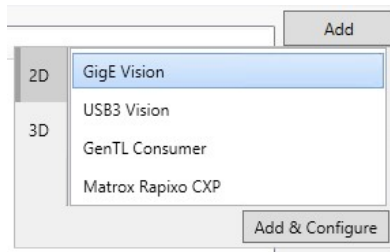
New projects and nearly all existing projects have at least one **Camera** step. Each **Camera** step specifies a camera source, which is configured by selecting a camera component defined in the **Platform Configuration** dialog.

Each camera component maps to a physical camera and has its own unique configurable properties. The act of mapping a camera component in Matrox Design Assistant to a physical camera is called camera allocation.



By default, a new project set to run on a PC runtime platform will try to allocate a camera of the type specified when the project is created. If you have a single dedicated camera on your network, then no special action is required for allocation. When there are multiple physical cameras on your network, but only a single camera component in your project, you must select which physical camera to allocate to your camera component. For more information, see the **Identifying cameras and changing the default allocation mode** subsection of this section.

If you want to use multiple cameras in your project or you want to add an additional 3D sensor, you can add these components using the **Platform Configuration** dialog. To add a camera component, click on the **Add** button in the **Cameras** page of the **Platform Configuration** dialog, and then select the supported camera or 3D sensor from the presented dialog. Click **Add & Configure** to specify configuration settings. Any previously added cameras will remain available.



It is recommended to test a new GenICam compliant camera using Matrox Capture Works, discussed below, before connecting to Matrox Design Assistant. In Matrox Capture Works, you can more easily diagnose and troubleshoot grabbing or connectivity issues. For more information about configuring GenICam feature settings, see the [GenICam feature settings for GigE/USB3 Vision or CXP cameras](#) subsection of the [Platform settings for your camera](#) section in [Chapter 32: Acquisition](#). For more information on using 3D sensors, see [Acquisition with 3D sensors](#).

▼ Matrox Capture Works

Matrox Capture Works, accessed through Mil Control Center - General Tools, handles discovery, configuration and viewing of GigEVision, USB3Vision, and GenTL devices. Using Matrox Capture Works, cameras implementing GigEVision or GenTL will be discovered, their IP address can be configured, and their Features can be accessed through the Feature Browser.

To launch Matrox Capture Works from the **PhysicalCameraN** page of the **Platform Configuration** dialog, click the **Open Capture Works** button.

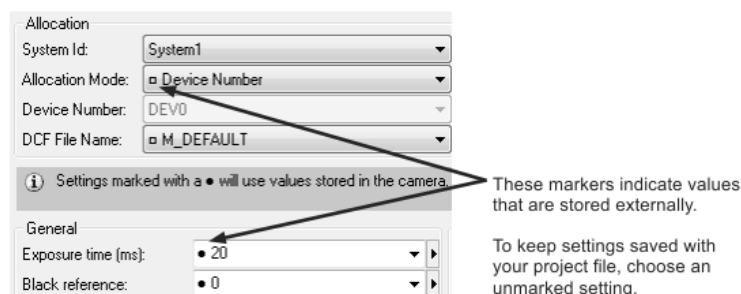
▼ Using the existing Matrox Design Assistant templates and example projects

When you create a project, you specify your own camera and system (typically, the system is added automatically). However, templates and example projects in Matrox Design Assistant are set to run using default settings. For example, the default camera type is GigE Vision. When you try to run a template or example project that grabs images, you might receive an error because the project attempts to access the default settings. To fix the problem, use the **Cameras** page of the **Platform Configuration** dialog to remove the listed camera component and add the camera that you are using.

▼ Using external defaults in your project

On the **PhysicalCameraN** page of the **Platform Configuration** dialog, little square markers indicate values that are stored in the runtime platform's registry. These are default values established outside of Matrox Design Assistant using the MILConfig utility.

It is strongly recommended to choose settings not marked with the little square. When you choose an unmarked option, your choice is saved in the Matrox Design Assistant project file. This keeps important settings tied to the project itself, rather than to external defaults. For example, the camera **Allocation Mode** is best set to the unmarked **Camera Name**, **IP Address**, or **Device Number** option.



Similarly, some camera settings are stored on the physical camera itself. A small black dot beside a setting on the **PhysicalCameraN** page indicates that the project will use the value stored on the camera. Again, it is recommended to set values without the dot. This ensures predictable project performance, even if you swap out a camera. For example, if you want to disable triggering for the camera, and ensure the setting is saved with your project, set the **Trigger** input to **Disabled**, instead of **• Disabled**.

For more information on configuring a GigE/USB3 Vision camera (which can, for example, have additional features not listed in the **Platform Configuration** dialog), see the [Platform settings for your camera](#) section in [Chapter 32: Acquisition](#).

▼ Identifying cameras and changing the default allocation mode

To identify the camera to use, you can specify its rank among the available cameras on the network (device number), its IP address, or its camera name (note that you cannot specify such settings in emulation mode). On the **PhysicalCameraN** page of the **Platform Configuration** dialog, set the **Allocation Mode** to the mode you will use to identify the camera. **Device Number** is the initial setting and has the advantage of simplicity; you specify the camera by specifying its device number, which is the order in which it is detected. For example, by default, Matrox Design Assistant will connect to the first available camera that it discovers (device number 0). If you have a single camera plugged into your PC runtime platform, it will be found automatically. However, if multiple users share cameras on the network, there is no guarantee which camera (if any) will be the

first unused one.

Depending on the default device number (device 0), using the initial setting (**Device Number**) might not be a good idea. To avoid the problem of always connecting to a different camera on the network, change the allocation mode to **Camera Name** or **IP Address**. This will uniquely identify a specific camera so the default allocation will always select the correct camera, as long as it is currently available.

You can only set the allocation mode to **Camera Name** if your GigE/USB3 Vision camera supports a persistent user-defined name. If it does, you can use Matrox Capture Works to give the camera a meaningful name. To see the current user-defined name, hover the mouse cursor over the device in the **Devices Pane**. To set a user-defined name, specify the **Device User ID** in **Device Control** in the **Feature Browser** tab. A GigE/USB3 Vision camera must be available to change any of its feature values, including its camera name; if another project has it allocated, you will not be able to access it from your current project.

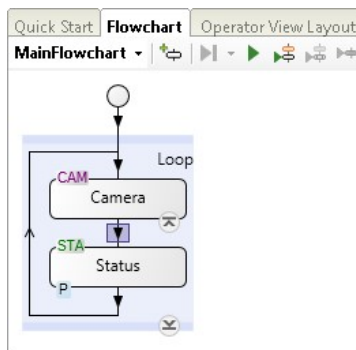
There can only be one default camera. If your project uses more than one camera, the mapping from camera component to a physical camera must be explicitly specified in the project.

For more information about allocating non-default or multiple cameras, or configuring the runtime platform for high volume image data, see [Chapter 32: Acquisition](#).

Building the flowchart


This section discusses the basics of building a flowchart, which involves adding and configuring steps. When you start a new project, you will be in the **Flowchart** tab, and you will see the default flowchart, which has a **Camera** step and a **Status** step within a loop.

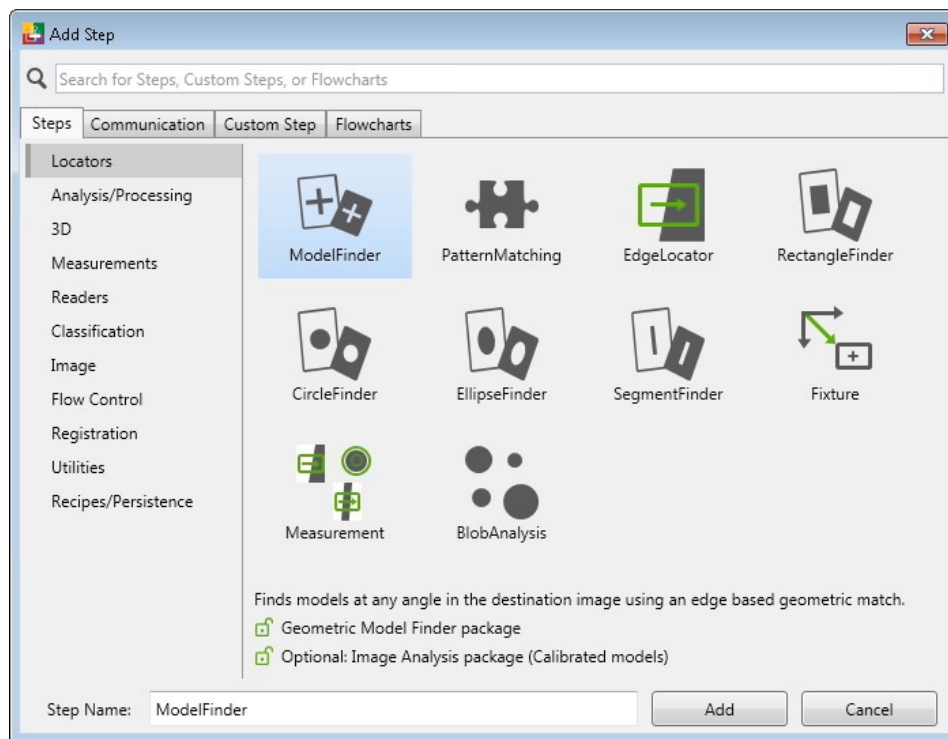
For a full list of steps that can be added and their categories, see the [Matrox Design Assistant steps and flowchart features](#) section in [Chapter 1: Introduction](#).



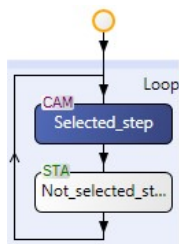
General procedure

To add and configure a step, perform the following:

1. Double-click on an [arrow](#) symbol in the flowchart (the step will be added below the arrow). The **Add Step** dialog opens. You can also open this dialog by clicking the **Add step** () toolbar button from the **Project** toolbar, or click on the **Add Step...** context menu item in the **Flowchart Step** context menu.



- Click on one of the following pages in the **Add Step** dialog:
 - Steps** page (selected by default). Allows you to select the step to add. Changing the name in the **Step Name** text box changes the displayed name of the step in the flowchart. You can find steps by selecting a tab from the list on the left that relates to your step, or use the search bar at the top to search for a step by name or description.
A green padlock symbol next to the package name indicates that the step has a permanent license, a yellow padlock indicates that the step has a provisional license (hover over the padlock to see when the provisional licenses expire), and a locked symbol indicates that the step does not have a license (and cannot be added to the flowchart).
 - Communication** page. Allows you to select a [communication step](#) to add. These steps control the information flow using a number of different protocols and hardware connections.
 - Custom Step** page. Allows you to select a [custom step](#) to add. Several custom steps are provided; some are for demonstration and training purposes (such as the **DateTime** custom step), and some are for real applications (such as the **OCR** custom step).
 - Flowcharts** page. Allows you to add a step that runs a previously saved [subflowchart](#), or to create a new, empty subflowchart.
- Click on the **Add** button. The **Add Step** dialog closes and your step (or subflowchart) is positioned in the flowchart. The order in which steps occur determines the order in which they are run. Typically, projects include multiple steps within a flowchart loop.
- Select the step (or subflowchart) and configure it using the [Configuration](#) pane. You can use the [Quick Access](#) pane to help guide your configurations. Note that in the flowchart, the colors of the selected step (or subflowchart) are inverted.



While building the flowchart, you will typically want to [run it and access its results](#). For more information on building your flowchart, see [Part 3: Flowchart control, expressions, and variables](#).

▼ Generally interacting with flowcharts

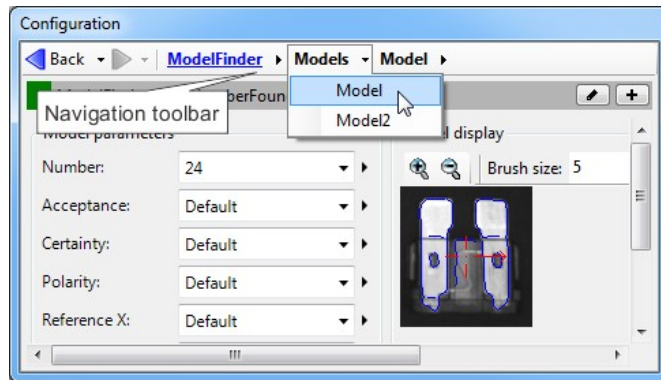
Basic interactions with flowcharts should be intuitive. This includes understanding flowchart symbols, printing flowcharts, dragging and dropping inputs or results (expressions), accessing the [Flowchart Step](#) context menu, and selecting, moving, and copying steps. These interactions can often be done by using a menu item (such as **View**), or by accessing parts of the flowchart itself (such as, right-clicking on a step, dragging and dropping it to a different location, or copying it with **Ctrl+C** and then pasting it in the new position with **Ctrl+V**). For more information about interacting with flowcharts, see the [Flowchart view](#) reference chapter.

▼ Using the Configuration pane

When you select a step, the [Configuration](#) pane displays the most important inputs that you can set. The [Configuration](#) pane also lets you configure an element in the [Operator view layout](#) tab.

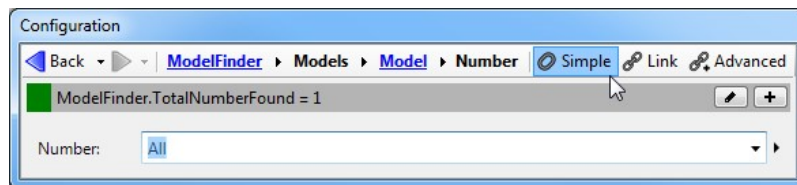
In addition to inputs, the **Configuration** pane has the **Navigation** toolbar, which consists of:

- **Navigation buttons.** Once you have displayed several inputs in the **Configuration** pane, you can use the **Navigate backward** (◀ Back) and **Navigate forward** (▶) toolbar buttons to cycle through them, even if they relate to a different step. The selected step on the **Flowchart** tab will automatically change to match.
- **Navigation links.** When an input is displayed alone in the **Configuration** pane, the navigation options will display links to related inputs and a link to the default **Configuration** pane for the associated step. When the inputs displayed belong to a member of a collection (for example, a set of models), a dropdown list next to the name of the collection lets you move quickly to one of the other members.



These inputs are also accessible in the **Properties** pane. Clicking on any of the inputs in this pane opens the **Configuration** pane with that input.

- **Simple, Link, and Advanced editor buttons.** These buttons are on the top-right of the navigation options when you are editing an input. Clicking on these buttons opens an editor where you can specify a value for the selected input. Click the **Simple** button to specify an explicit value, click the **Link** button to specify a value that is a link to another input, and click the **Advanced** button to specify a value that is an expression. The following is an example of clicking the **Simple** button to specify an explicit value.



The editor buttons are only present when configuring a single input. When multiple inputs are displayed in the **Configuration** pane, you can access the **Link** and **Advanced** editors by clicking on the **Alternate options** (⌵) button. For more information on using the simple, link, and advanced editors, see the [Configuration pane and editors](#) section in the [Panels and editors](#) reference chapter.

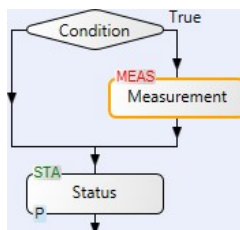
When you select an analysis step, the **Configuration** pane will display the **Condition bar**, which allows you to access the **Condition** editor.

▼ Using the Quick Access pane

The **Quick Access** pane is your guide to configuring each step in your project. This pane consists of instructions, hints, tips, and links to inputs in the **Configuration** pane of the selected step. Note that some advanced inputs might only appear in the **Configuration** pane when they are accessed using the **Quick Access** pane (or by the **Properties** pane, which lists all of the selected step's inputs).

▼ Flow control steps

Matrox Design Assistant has several steps capable of altering the flowchart's flow. These steps can, for example, create loops, delay the flowchart's progression, and use a condition to create multiple paths, all depending on runtime events. For **Flow Control** steps, you enter an expression that evaluates when to execute a sequence of steps. The following example shows how to control the flow with a **Condition** step. If the condition is true (such as a model being found in an image), the **Measurement** step is run; otherwise, the flowchart runs directly to the **Status** step.



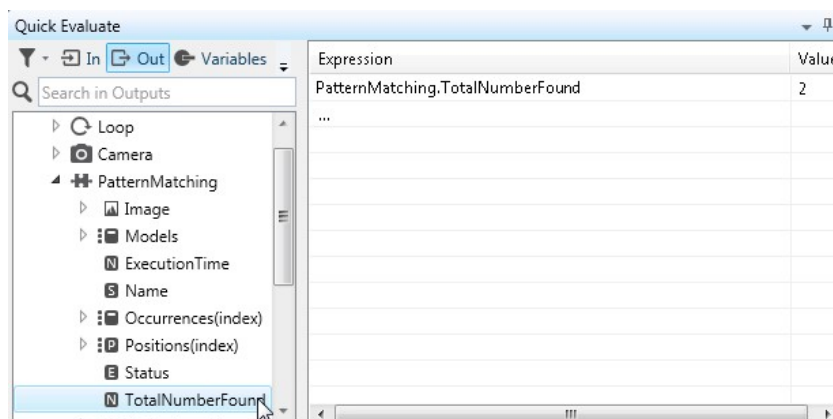
Flow Control steps are in the **Flow Control** section of the **Add Step** dialog; they include: **Condition**, **Break**, **Switch**, **Continue**, **Halt**, **Loop**, **Status**, and **Error**. For more information, see [Chapter 27: Flow control steps](#).

▼ Variables

Matrox Design Assistant lets you use [variables](#) to store a specified value (or image) that can change. Variables can be useful when, for example, writing [expressions](#), or directly [binding](#) an input element from the operator view of a running project. There are 16 possible types of variables, which include **Numeric**, **String**, **Boolean**, **Image**, **Fixture**, **Step**, and **Timestamp**. To store a collection of values (such as a group of numbers, strings, or objects), you would specify an [array type variable](#), such as an **Object array** or a **Numeric array**. For information on arrays, see [Chapter 31: Using arrays](#). For information on other variables types, see the [Data types](#) section in [Appendix A: Expression syntax](#).

▼ Expressions

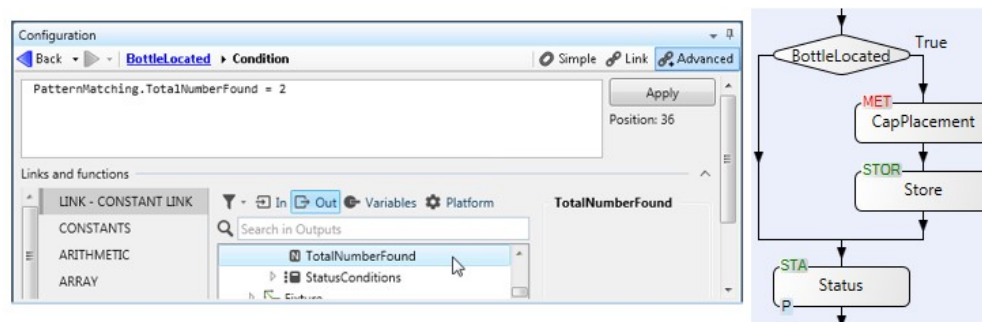
Expressions are statements that evaluate information and set or return a value. You can use expressions to set a step's input or a variable to a value, provide a **Flow Control** step with a conditional statement, calculate an output value of a step for a log file, or return a simple result. For example, to get the total number of occurrences found by the **PatternMatching** step, you can write the expression, `PatternMatching.TotalNumberFound`, in the **Expression** column of the **Quick Evaluate** pane; in the following case, the expression returns a value of 2.



Although expressions can be explicitly written, you can create them interactively; this is the recommended way so that you can avoid syntax errors. You can typically create basic ones by accessing the related [tree structure](#). In the above example, double-clicking on the **TotalNumberFound** output generates the expression. Alternatively, you could drag the **TotalNumberFound** from the **Quick Watch** flyout panel.

Expressions range from simple to complex and can be composed of links to step inputs and outputs, functions, operators, and variables. You will typically specify expressions in the **Configuration** pane (to configure a step) and the **Quick Evaluate** pane (to retrieve a result). It is common to test and evaluate expressions (in the **Quick Evaluate** pane) while building them in their final destination (a step's **Configuration** pane). Note that you can drag and drop expressions across panes. The expression column and the [tree structure](#) of the **Quick Evaluate** pane can both be sources for your drag and drop.

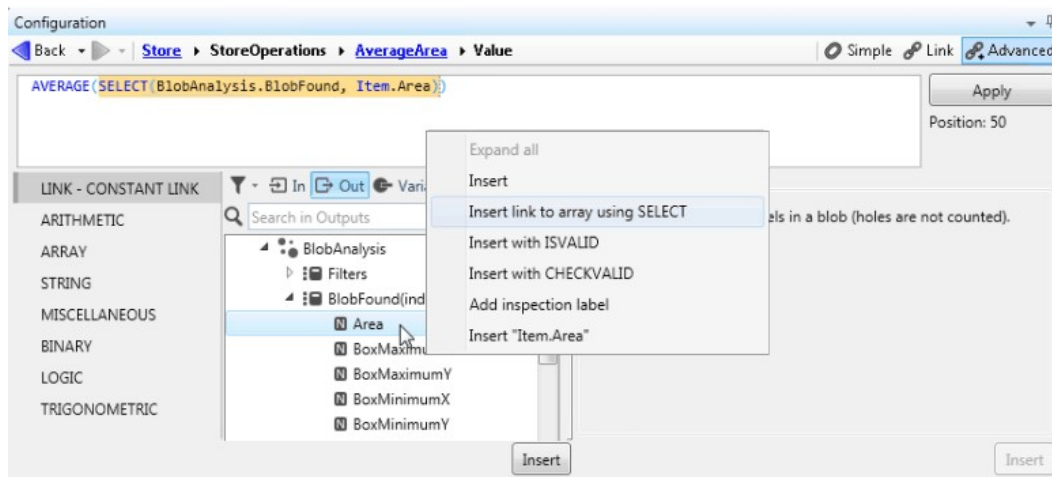
The following example shows an expression, from the **Configuration** pane of a **Condition** step that has been renamed **BottleLocated**. The expression uses a link to the **TotalNumberFound** output of the **PatternMatching** step to determine whether the required number of occurrences was found. Since this is part of a condition (**BottleLocated**), the subsequent steps are only executed if the expression is true.



If you need a complex expression, you will typically use the **Configuration** pane's **Advanced** editor. Complex expressions generally have multiple parts that often include [functions](#), which return a value when given inputs (for example, the arithmetic function `SQRT`). There are many categories of functions, such as array-related functions; these can be useful when dealing with steps, such as the **BlobAnalysis** step, that have collections of inputs and outputs, which you must access using arrays. For example, the following expression uses 2 array functions, `AVERAGE` and `SELECT`, and a collection type output from the **BlobAnalysis** step. The following complex expression iterates through the collection of blobs found and returns their average area as a single number:

```
AVERAGE(SELECT(BlobAnalysis.BlobFound, Item.Area))
```

You can also use the right-mouse menu options to insert [sub-expressions](#), which is useful when writing an expression with arrays. For example, to write the expression above, you can select `AVERAGE` from the **ARRAY** tab. You can then right-click on the **Area** output of the **BlobAnalysis** step and insert link to an array using `SELECT`, shown in the example below.



For more information about expressions, see [Chapter 30: Building expressions](#).

▼ Subflowcharts

All flowcharts in a project, other than the main flowchart, are considered subflowcharts. You can either [add a subflowchart as a step](#) or [execute a subflowchart from an event](#).

Flowcharts can become long and difficult to follow; subflowcharts added as a step to a flowchart can help to better organize similar parts of a project, and can be used across multiple projects. For example, you can have a subflowchart that handles communication with a programmable logic controller (PLC), which is often developed by someone other than the vision specialist. You can then add this subflowchart as a single separate step to your main flowchart or even to another subflowchart. Note that, to inspect a variant of a family of related products, you should use [recipes](#).

Subflowcharts can be called at runtime as a result of an event occurring. For example, clicking a button from the operator view can call a subflowchart that lets the user redefine a new match model or switch recipes.

▼ Recipes

Recipes are a group of step inputs and variables used to inspect multiple variants in a family of related products. With recipes, you can manage product variations without duplication and without having to change what is common. The following is an example of a project that would use recipes (named Banana, Blueberry, and Strawberry) to inspect yogurt lids. Many aspects of this project would be common (for example, PLC communications, lid diameter, operator view publishing), while other aspects (handled by each recipe) would be different (for example, label color and status condition).



One product, three variations, three recipes

For more information about what recipes are and how to use them, see [Recipes](#).

▼ Saving images and text

To save an image, use the [ImageWriter](#) step. You can save the full image or a part of it, and have the step annotate it directly. You can also save a [sidecar file](#) with information relating to the saved image (such as setup conditions or results per image). Sidecar files can store information as key-values (**KeyValue** text mode) or as plain text (for legacy purposes).

To generate and save text files (such as logs, reports, and configuration files), use the [TextWriter](#) step. Files can be organized by key-values, in CSV format, or in plain text (for legacy purposes). Whenever possible, it is recommended that you use key-value or CSV text, since DA automatically protects against unavailable links when doing so. DA provides you with tabular configuration panes to simplify adding this type of structured text.

You can save all files (images or text) locally on your runtime platform or on a shared network drive.

▼ Custom steps

A **Custom** step is a step that extends the capabilities of a project by allowing you to add your own code. This is typically done for proprietary communication protocols, proprietary processing algorithms, or for accessing MIL functions that are not available in Matrox Design Assistant.

You can make a new **Custom** step or use the ones provided. The **OCR** custom step can be a faster alternative to the **StringReader** step. The **DateTime** custom step, **ArrayInput** custom step, and **ImageRotator** custom step are intended to demonstrate how custom steps work. Do not use them in an actual project; there are better alternatives for the functionality that they provide.

To create or import a new **Custom** step, see the custom step documentation. This can be found, along with the source code and DLL files for the provided **Custom** steps, by clicking on the **Custom Step** button in the Matrox Design Assistant **Quick Start** tab.

Custom steps do not provide previews of their results. You must rerun the step to see the effect of changing settings. Custom steps do not put results in the **Results** pane. To view results, use the **Quick Evaluate** pane.

Techniques for running the flowchart and accessing results

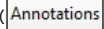
Developing a project typically involves running the flowchart at design-time from your development computer, viewing results, and modifying settings. This process is repeated until the flowchart produces the results you expect. The following subsections list some of the techniques you can use to help with this process. Each subsection links to the [User Interface Reference](#), where you can get more information. Techniques are listed according to how often they are typically used. These techniques can also help testing. For more information on tips to help testing and debugging, see the [Testing and debugging a project at design-time](#) section later in this chapter.

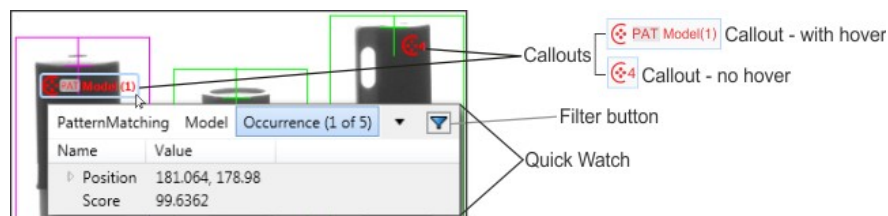
▼ Debug buttons

In the **Project** toolbar, there are several toolbar buttons that let you perform general debugging actions, such as running the project to a selected step. These buttons are also available from the **Debug** menu.

	Reset	Ctrl+Shift+F5
	Run	F5
	Run to Selected Step	F6
	Next Step	F10
	Step Into Next	F11
	Rerun the Selected Step	F9
	Leave Quick Run Mode	
	Create Quick Run Group	
	Remove from Quick Run Group	
	Display the Running Step	
	Stop at inspection end	

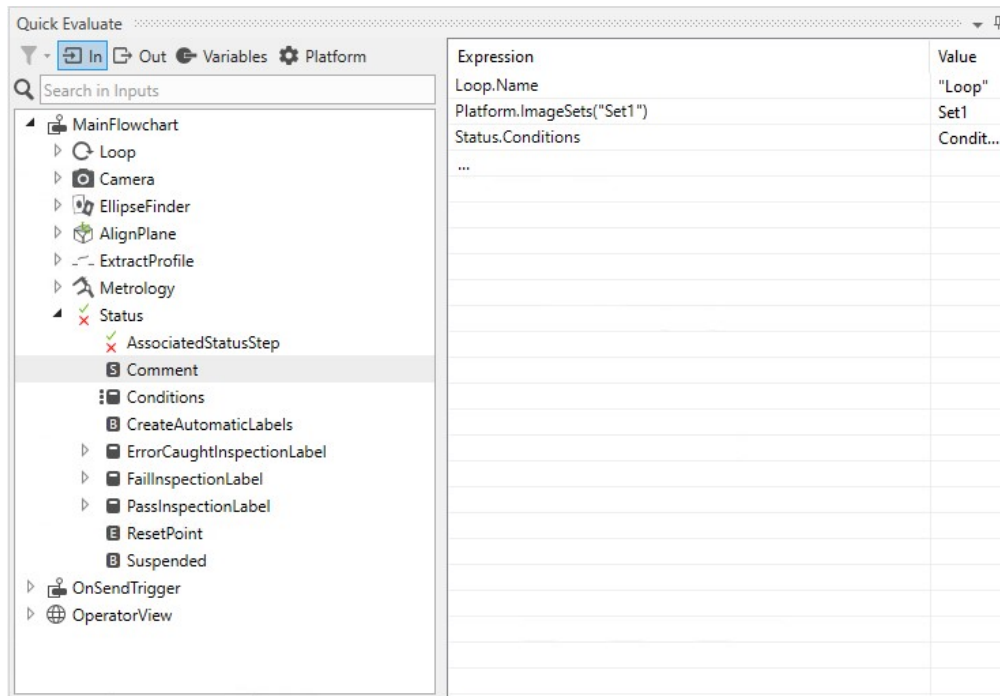
▼ Quick Watch flyout panel and callout annotations

The **Quick Watch** flyout panel and callout annotations let you easily access key results from the selected step's displayed image. To open the **Quick Watch** flyout panel, click on a callout annotation, which is a small graphic that is typically visible for each result of the selected step's displayed image. Hovering on a callout reveals more information. You can drag and drop results from the **Quick Watch** flyout panel into other parts of your project (for example, into the **Quick Evaluate** pane or the input of another step). To manage how callouts are displayed (visibility), click the **Annotations** () toolbar button in the **Project** toolbar.



▼ Quick Evaluate pane

The **Quick Evaluate** pane lets you navigate to, and display the current value of any step's input or output/result, as well as any variable or platform configuration setting, while a project is running. It also allows you to evaluate images, by viewing one or more simultaneously. You can use this pane to add, modify, drag and drop, and copy and paste expressions.



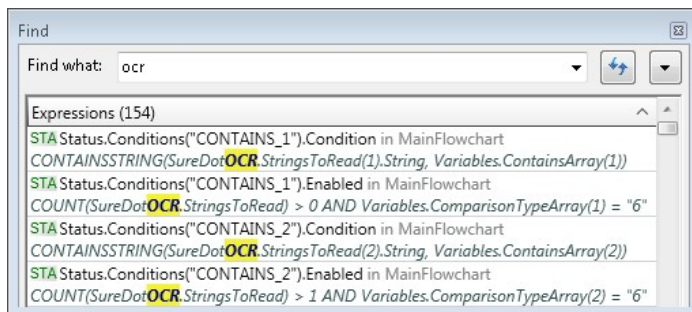
▼ Camera Images pane

The **Camera Images** pane manages and displays which images each **Camera** step uses (including image sets). The top-left of this pane has a **Filters** option that lets you select the images to display, based on whether they have a pass or fail status.



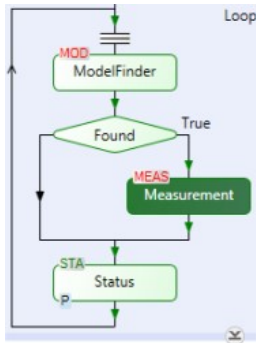
▼ Find pane

The **Find** pane, accessible from the **Edit Find** menu item (or **Ctrl+F**), lets you search a project for the specified search query.



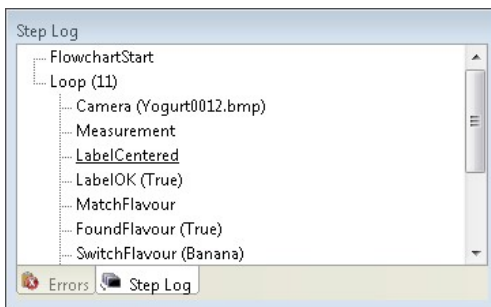
▼ Quick Run mode

When you are in **Quick Run** mode, only the selected step, and the steps it depends on, are run. when clicking the **Run to selected step** (▶) toolbar button does not reach the step you want. To use Quick Run, right-click on the step to test and choose the **Create Quick Run Group** context menu item. You are now in Quick Run mode and the flowchart is analyzed to determine all the steps upon which the selected step, also known as the root step, depends. The root step, and the steps it depends on, will be in green. All other steps are hidden with a collapse symbol.



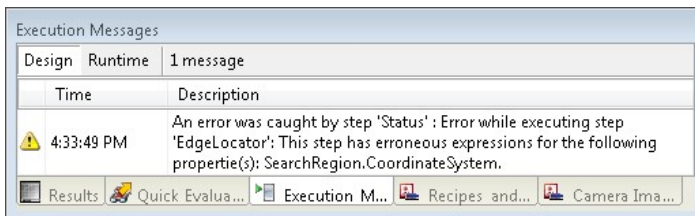
▼ Step Log pane

The **Step Log** pane displays information about the execution of your steps, such as the step on which your project stopped, as well as the number of times each loop in your project was performed.



▼ Execution Messages pane

The **Execution Messages** pane displays all the messages generated during design-time or runtime (as specified with the tabs at the top of the pane).



▼ Results pane

The **Results** pane lets you access results. Results are also easily accessible from the **Quick Watch** flyout panel (this is the recommended way to access results).

	Model	Occurrence Index	Score	X	Y	Angle	Execution Time
▶	Model	1	98.7272	156.001	291.738	0	0.0008256 s
	Model2	1	99.1964	466.485	293.866	0	
	Total found: 2						

Cameras and images



Matrox Design Assistant grabs, corrects, processes, and analyzes images, which you can acquire [from a camera](#), a [3D sensor](#), or an [image set](#). To specify related settings, use the **Platform Configuration** dialog.

To manage image files, use the **Camera Images** pane.

▼ Camera step

There are several sources of images that Matrox Design Assistant can use, but the typical source is the **Camera** step. The **Camera** step's main function is to fetch an image and pass it to whichever steps need it. Image data is typically acquired from physical cameras, possibly in response to a trigger. The **Camera** step will wait until an image is available. The **Camera** step can also simulate a grab by passing along images from the network or local disk (image set), rather than images from your camera.

Every typical project needs at least one **Camera** step to work with images, and so, by default, every new project starts with a single **Camera** step in a loop.

The **Camera** step's display view allows you to view the image being acquired. When acquiring images from a camera, the display view allows you to adjust the lighting, aperture, and focus of the physical camera without having to loop through the flowchart. Use the **Camera Live** ( **Camera Live**) toolbar button in the **Project** toolbar to see the camera's real-time view. Note that this is only available if triggering is not enabled on the camera. When the **Camera** step is first added to the project, or the project is first opened and the step has not run, the display view for the **Camera** step is blank. To view the image being acquired, select the **Camera** step and click on the green triangle in the display view, or click on the **Run to selected step** () toolbar button in the **Project** toolbar.

When using a GigE Vision camera or a USB3 Vision camera, your camera must be connected, turned on, and be accessible through your network or directly connected to your runtime platform. It is extremely important that your network be configured appropriately to handle the large volume of data that can be generated by GigE and USB3 cameras. You can use the Matrox Capture Works utility to help configure a new GigE or USB3 camera. See the [Matrox Capture Works](#) subsection of the [Cameras and your runtime platform](#) section earlier in this chapter for more information.

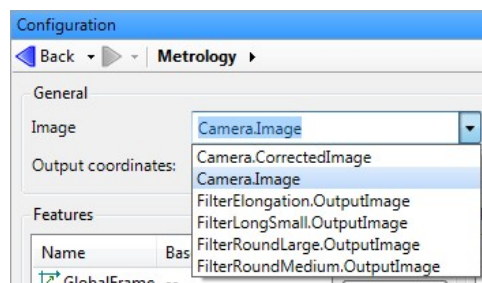
The **Camera** step also provides images of the appropriate bit depth to steps that link to the **Camera** step. When enabled, the **Camera** step's remapping settings specify what portion of the input data values to keep, and convert the data to the right bit depth as needed.

The **Camera** step is discussed in more detail in the [Procedure for using the Camera step](#) section in [Chapter 32: Acquisition](#).

▼ Analysis steps and the image they use

Analysis and Processing steps, such as the **BlobAnalysis** step and the **SureDotOCR** step, usually come after the **Camera** step, and typically operate on images grabbed using the **Camera** step.

Some steps, such as the **ImageProcessing** step, do not only operate on images, but also create images for use by other steps. Consequently, every step that uses an image requires that you select the image on which to operate from the **Image** property, configurable in the **Configuration** pane. The format for specifying an image is `StepName.ImageName`. Keep in mind that some steps can output different images, and so are specified accordingly. For example, the **Image** property's dropdown list shown below offers 2 separate **Camera** step images: `Camera.Image` and `Camera.CorrectedImage`. The 4 other images in the dropdown list are outputs from separate **BlobAnalysis** steps.



Only select the **Camera.CorrectedImage** option if the **Camera** step is associated with a calibration file.

Sometimes, it is useful or necessary to use images stored on disk rather than grabbed from your camera. Typically, disk images are used when you are building or testing a project and you want to use either an ideal image or an image with a known defect. Note that when in emulation mode, you can only use images on disk. For more information on accessing images from disk, see the [Procedure for configuring image sets](#) section in [Chapter 33: Acquisition from image sets](#).

You can also use static images, which are taken from the image repository on your runtime platform and are used as master images or templates by some steps. Any step that uses images can use a static image. For more information about static images and the image repository, see the [Using static images](#) section in [Chapter 37: Static images](#).

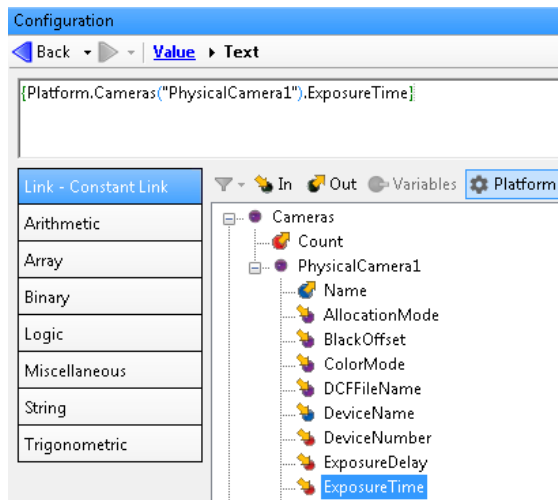
For information on supporting color images, see the [Dealing with color images](#) section later in this chapter.

▼ Initial platform settings

To adjust the various settings required to grab an image, including the triggering options discussed below, use the **Platform Configuration** dialog. The default settings of the **Platform Configuration** dialog are sufficient for most projects. In this dialog, you can adjust the exposure and triggering properties, among others. See the [Platform settings for your camera](#) section in [Chapter 32: Acquisition](#) for information on the various options available.

Often, after testing the initial setup, you want the selected settings to remain unchanged; however, when you require more than one grab configuration (for example, one configuration for training and another for inspecting), it is necessary to make runtime changes. To make camera-related runtime changes, you can [bind input elements](#) directly to platform configuration settings or use the **CameraSettings** step. You can also link a [Button element](#) to the **Change Camera Settings** action. For more information, see the [Changing the acquisition settings at runtime](#) section in [Chapter 32: Acquisition](#).

To include any of the platform configuration settings in an expression, you can link to them. For example, to link to the camera's exposure time while working in the operator view, use the **Configuration** pane to find the setting in the tree structure, and apply the link.



▼ Setting up your physical camera

You can acquire images in Matrox Design Assistant by attaching a camera to your runtime platform (if not using a smart camera), adding a **Camera** step to the project, and, if necessary, initializing your camera settings to set the grabbing trigger.

The procedure for configuring a (triggered) grab depends on your hardware configuration. Common setups are described below. Note that the following procedures assume that the correct camera type is already specified and allocated.

▼ Using a Matrox smart camera with no trigger

To use a Matrox smart camera in continuous mode (with no trigger), perform the following:

1. Open the **Platform Configuration** dialog.
2. On the **PhysicalCameraN** page, ensure that the **Trigger** property is disabled, and click on the **OK** button.

▼ Using a Matrox smart camera with a hardware trigger

To use a Matrox smart camera with a hardware trigger, perform the following:

1. Using the description of the pinout of your Matrox smart camera, determine to which of the auxiliary input signals you will connect the trigger source. For example, with Matrox Iris GTR/GTX, you can connect the trigger to auxiliary input signals 3, 4, 5, or 6; typically 3 is used for triggers. Refer to the *Installation and Technical reference* of your Matrox smart camera for the pinout of the connector.
2. Open the **Platform Configuration** dialog.
3. On the **PhysicalCameraN** page, enable the **Trigger** property and then select an auxiliary input signal from the **Trigger source** property's dropdown list. Note that the selected triggered signal will not appear in the **I/O** tab, unless that input signal is also being used for other purposes (see the [Matrox Advanced I/O Engine](#) section in [Chapter 52: IO steps](#)).

▼ Using a GigE/USB3 Vision camera with no trigger

To use a GigE/USB3 Vision camera with no trigger, perform the following:

1. Open the **Platform Configuration** dialog.
2. On the **PhysicalCameraN** page, ensure that the **Trigger** property is disabled, and click on the **OK** button.

▼ Using a GigE/USB3 Vision camera with a direct hardware trigger

To use a GigE/USB3 Vision camera with a direct hardware trigger, perform the following:

1. Refer to your camera's documentation to determine which pin on your camera's connector receives the trigger signal.
2. Open the **Platform Configuration** dialog.
3. On the **PhysicalCameraN** page, enable the **Trigger** property, and then select an input signal (line) from the **Trigger source** property's dropdown list.

If you need to delay the grab relative to the trigger, for a Matrox smart camera or a GigE/USB3 Vision camera, see the [Using an input to automatically generate a delayed pulse on output](#) subsection of the [Matrox Advanced I/O Engine](#) section in [Chapter 52: IO steps](#).

▼ Initiating a grab

There are several ways to initiate a grab, depending on the options that you choose in the **Platform Configuration** dialog and how you set up both your flowchart and your runtime platform. By default, when the flowchart arrives at the **Camera** step, a grab is initiated, activating a strobe if one is set up. Subsequent steps can use this grabbed

image (`Camera.Image`) until a new grab is initiated, usually in the next iteration of the main loop.

The alternate ways to initiate a grab are based on signals from external devices, as well as from flowchart controlled events and time intervals. For a more complete explanation of how to set up and when to use these alternate ways to initiate a grab, see the [Triggering](#) section in [Chapter 32: Acquisition](#).

▼ **Links to more information**

Matrox Design Assistant supports analysis and processing on various image types. For information on handling images with bit depths greater than 8-bit (for example, mono10 or mono12), see the [Displaying images with more than 8-bits](#) section in the [Display view](#) reference chapter, or see the [Remapping](#) subsection of the [Procedure for using the Camera step](#) section in [Chapter 32: Acquisition](#). If your project uses depth maps, see the [Procedure for generating a depth map](#) section in [Chapter 42: Working with 3D data](#).

For information on using more than one physical camera (in one or more [Camera](#) steps), or on specifying a specific camera, see the [Using multiple cameras](#) section in [Chapter 32: Acquisition](#).

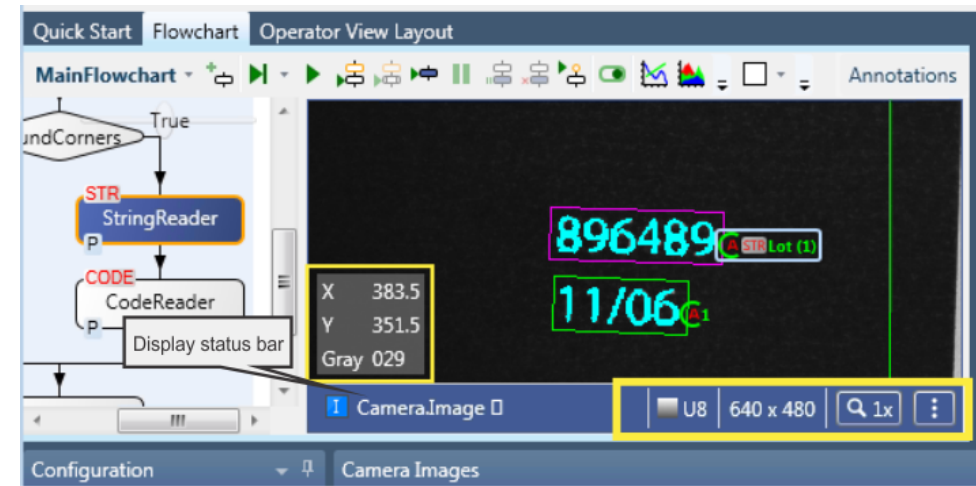
Display features during design-time

There are several display-related features that you can use to help develop your project, such as the display status bar, annotations that help you manage results, side by side display panes, 2D or 3D views, and graphs that show you image statistics.

By default, when you access the [Flowchart](#) tab in the [user interface](#), you see both the flowchart and the selected step's displayed image.

▼ **Display status bar**

The display status bar, located below the selected step's displayed image, displays the image depth, image type, and image dimensions. The X- and Y-coordinates (and Z-coordinates in the case of depth maps) and the gray intensity or color value at the position of your mouse cursor as it passes over the image (the zoom factor is also shown) is displayed in the bottom-left corner of the display view. The coordinates are in calibrated world units if the step's image is associated with a calibration; to show them in pixel units, click on the **Display options** menu in the display status bar and uncheck the **Calibrated Coordinates** context menu item.

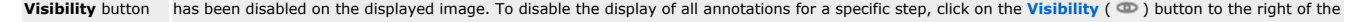




▼ **Displaying a step's images**

You can choose an image to display from the various images associated with a step or choose a display image from the **Display options** menu for steps without built in images. You can also view multiple images for a selected step in the display view. For more information, see the [Displaying a step's images](#) section in the [Display view](#) reference chapter.

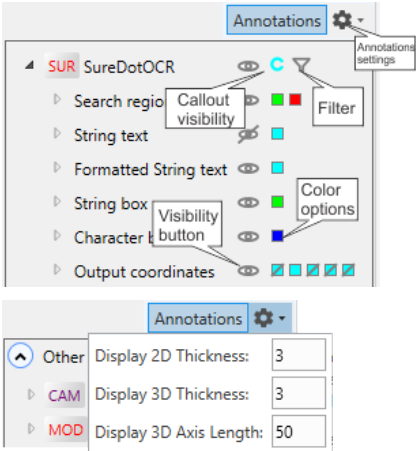
▼ **Annotations**

You can modify whether annotations are hidden, or how they are displayed, using the [Annotations](#) control panel. This is opened by clicking on the [Annotations](#) toolbar button in the [Project](#) toolbar. The [Annotations](#) control panel provides a list of all the annotations in the project, grouped according to the steps to which they belong. Use the following buttons to modify the annotation settings.

Button	Description
Annotations settings toolbar button	Opens a dropdown tab with 2D Thickness , 3D Thickness , and 3D Axis Length from the Annotation groups toolbar. For more information on design-time annotation thickness, see the Annotations section in the Display view reference chapter
Annotation Filter button	Filters the list of available annotations. By default, the key annotations for the selected step are listed. Click on this button to view all the available annotations for the selected step; it will have a line through it, indicating that filtering has been disabled.
Visibility button	Specifies whether or not to display a specific annotation on the step's displayed image. A line through this button indicates that the associated annotation has been disabled on the displayed image. To disable the display of all annotations for a specific step, click on the Visibility () button to the right of the step name.
	Specifies whether callouts are always visible or hidden. When displaying annotations from other steps, their callouts will typically appear only when the


Callout visibility button	cursor hovers over these annotations. You can set how to display callouts for a specific step by clicking on the Callout visibility () button to the right of the step name. The typical default behavior is to only display the annotations of the currently selected step. The Status step displays all the callout annotations for every step. For more information, see the Using the Quick Watch flyout panel and callout annotations section in the Display view reference chapter.
Color options button	Assigns colors to different states of the annotation. Each annotation has one or more color options to the right of its Visibility () button. By default, an annotation appears green for a successful operation, and red for a failed operation. For example, when locating measurement markers, the annotation appears green when a marker is found, and red when a marker is not found. If an annotation does not automatically indicate pass or fail with green or red, or if you want to customize the color options, see Setting annotation color .

Note that you can modify annotations for each display pane when working with multiple display panes.



For more information on design-time annotations, see the [Annotations](#) section in the [Display view](#) reference chapter.

▼ Display layout

You can change the display layout by specifying the layout you want from the **Select display layout** () toolbar button in the **Project** toolbar. You can choose between **Single**, **Dual H** (which specifies a side by side view), **Dual V** (which specifies a stacked view), and **Quad** to select the layout you want. This is useful for comparisons of sources and destinations or input and output images. When you have multiple display panes, the selected display pane is outlined in blue and it is the only display pane that will be affected by annotations and drawings. All operations have a default display layout.

▼ Display graphs

There are 3 graphs that convey pixel data about the selected step's displayed image: the [Line Profile](#) graph, the [Histogram](#) graph, and the [Edge Values](#) graph.

In the **Project** toolbar, you can access these graphs by clicking on their respective toolbar buttons . The graphs are for design-time and do not directly impact the project. They are intended to provide useful information for setting up the project. For example, you can use the [Line Profile](#) graph to draw a line across a dot that makes up a dot-matrix character. This lets you determine where the pixel intensity shifts, from bright to dark and from dark to bright. You can use the distance between these major intensity shifts to establish the dot diameter to use for performing character recognition with the [SureDotOCR](#) step.



▼ 2D and 3D views

When working with 3D data, you can choose to display the 3D view or 2D view of the image from the display status bar. For more information about 2D and 3D views, see the

[3D display options](#) subsection of the [Displaying 3D data](#) section in [Chapter 42: Working with 3D data](#).

▼ Zooming

You can zoom the flowchart using the [sliding zoom bar](#), which fades in more clearly as you hover over the top-right of the flowchart. To zoom the selected step's displayed image, use the mouse wheel or adjust the zoom factor in the display status bar. Note that zooming only occurs in the selected display pane when working with multiple display panes.

▼ Displaying images with more than 8-bits

For images with more than 8-bits per band, you can select a **View mode** to produce an 8-bit displayable image. For more information on the available view modes and how they display images, see the [Displaying images with more than 8-bits](#) section in the [Display view](#) reference chapter.

▼ Displaying images using a colormap

Matrox Design Assistant provides predefined colormaps that allow you display the image of a step in pseudo-color. A full description of all the available color maps is available in the [Displaying images using a colormap](#) section in the [Display view](#) reference chapter.

Real-world units and fixtures

In Matrox Design Assistant, you are able to calibrate your camera and fixture objects, allowing you to, for example, consistently place search regions and retrieve results.

▼ Calibration

Once an image is acquired, it can have a calibration file associated with it that allows other steps to return data in real-world units, rather than pixels. For instance, the [Measurement](#) step can return the length of an object in either pixels, without any calibration, or in mm or cm with an associated calibration file. A calibration file can also be used to correct different types of image distortion.

To create a calibration file, you most commonly use the [Matrox Design Assistant configuration](#) portal web pages, but you can also add a [Calibration](#) step to the flowchart. For more information on calibration, see the [Calibration](#) section in [Chapter 34: Calibration](#).

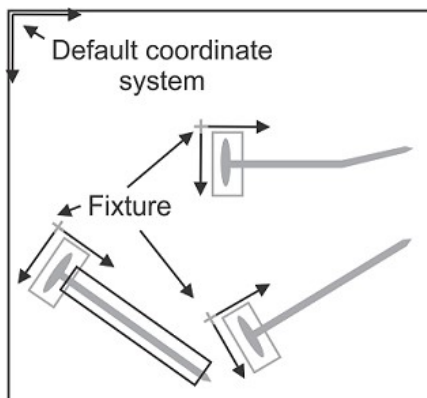
Once you have a calibration file, you can associate it with an image using the [Camera](#) step. All subsequent steps that use this calibrated image are able to return values in real-world units. The [Camera](#) step can also use the calibration file to create a corrected image, `Image.CorrectedImage`, which can in turn be used by any other steps. The [CalibrationName](#) and [CorrectImage](#) properties can be set in the [Configuration](#) pane of the [Camera](#) step. See the [Procedure for using the Camera step](#) section in [Chapter 32: Acquisition](#) for more information.

▼ Fixturing

Matrox Design Assistant supports fixturing. A fixture defines a relative coordinate system and consists of a point of origin and an angle. Once a fixture has been created, any step can use this fixture.

Fixturing is often used when the objects that need analyzing can appear in different places in the image. For example, a [ModelFinder](#) step determines the location of the object in the image, and then creates a fixture based on the object's variable location and orientation. This allows the object to be in a different position for each image taken, while not affecting the search areas and regions of interest of the subsequent steps.

While search regions are usually fixtured, step position and angle results are, by default, produced with respect to the absolute coordinate system, which has its origin at the top leftmost point of the image or at the calibrated origin. By fixturing a search region, the region will be oriented with respect to the fixture's coordinate system instead of the absolute coordinate system. Although not normally done, you can also specify a fixture for the output coordinate system; in this case, any results returned by a step are relative to the fixture's coordinate system. Note that a step's search region's fixture and its output coordinate system's fixture can be different.



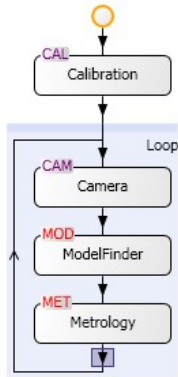
Matrox Design Assistant has several steps that can create fixtures. A project can have multiple distinct fixtures that are usable by other steps. Any search region that can use a

fixture can select one in the **Fixture** dropdown list located in the construction pop-up window. The construction pop-up window appears when creating or redefining a shape (in this case, a search region).

For more detailed information, see [Chapter 36: Fixturing](#).

▼ An example of calibration and fixturing

To measure the width, in mm, of an object that will appear in different places in sequential images, the following flowchart could be used:



The **Calibration** step appears first and outside of the main loop. Configure this step so that, for example, every 5 pixels is 1 mm; see the [Procedure for using the Calibration step](#) section in [Chapter 34: Calibration](#) for how to create a calibration file. Associate the image created in the **Camera** step with the calibration file created in the **Calibration** step; see the [Procedure for using the Camera step](#) section in [Chapter 32: Acquisition](#) for the correct procedure. Configure the **ModelFinder** step to find and fixture the object, regardless of the object's location in the image. Enter the fixture, created in the **ModelFinder** step, in the **Output coordinates** property of the **Metrology** step. This can be done by selecting the correct fixture from the list in the **Output coordinates** dropdown list, found in the **Configuration** pane of the **Metrology** step. Set the **Metrology** step to output the width of the object. See [Chapter 15: ModelFinder step](#) and [Chapter 20: Metrology step](#) for the appropriate procedures.


Search regions

When an **Analysis and Processing** step is processing an image, it can be beneficial to create a search region. A step that uses a search region will only consider what is inside the region, and ignore the rest of the image. This can speed up processing by eliminating unnecessary work. Using a search region can also make a step more robust, minimizing any unwanted noise that might make a target difficult to find.

A search region can be a rectangle, elliptical arc, polygon, or elliptical ring sector. Once a search region is created in a step, other steps can use it as well. Some steps can create a set of replicated search regions, with each region having same shape but occurring at a number of fixtures. Note that not all steps can create search regions; and those that can, are not necessarily able to create all search region shapes.

▼ Defining search regions

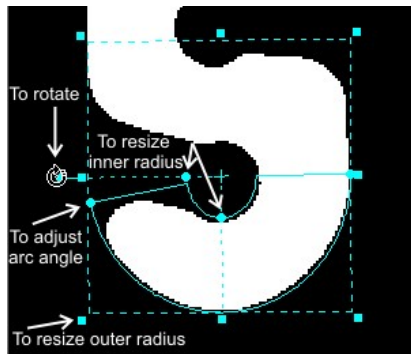
You can define a search region interactively on the selected step's displayed image, or by using the selected step's **Configuration** pane.

To create a search region interactively on the displayed image, click on the **Define a region**  toolbar button in the **Project** toolbar, and then click on the image to define the shape. The **Quick Access** pane should give you information about how to create the search region shape. Search region shapes can vary depending on the step you are using.

Whenever you create or redefine a shape interactively, a construction pop-up window will appear, as well as a yellow rubber band that connects the shape to its fixture. Set the shape's fixture (reference point and angle) in the construction pop-up window by selecting a fixture from the list.

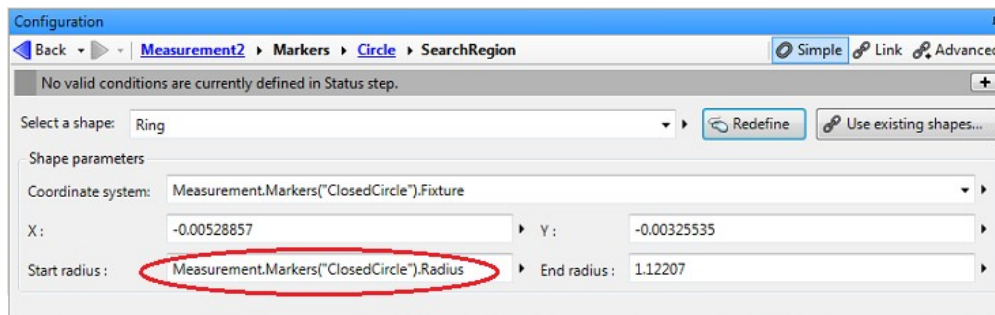
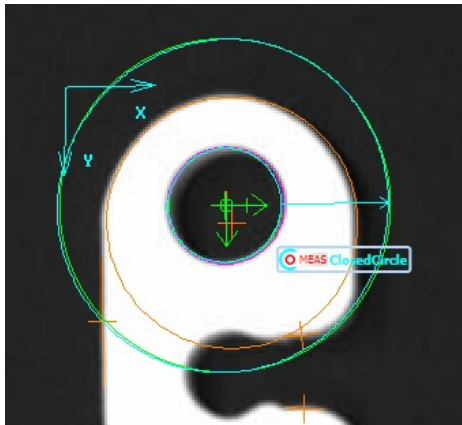
To edit or move an existing search region interactively, click on its edge in the image. A bounding box will appear, allowing you to resize and re-orient the image.


Search regions can also be defined relative to coordinate systems such as the **Absolute**, **Pixel**, **Display**, or **Tool** coordinate systems using the **Relative to** property within the selected step.



To use a search region defined in another step, click the **Use existing shapes** button in the selected step's **Configuration** pane. Use the tree structure in the **Link** editor to select the search region you want to use. The shared search region will be updated automatically based on any changes to the original.


You can specify the dimensions of a search region using explicit values or expressions in the selected step's **Configuration** pane. It is common to use expressions derived from the dimensions of earlier steps to define a new shape. You can drag and drop values from the **Quick Watch** flyout panel of other annotations directly into a dimension field of the shape that you are defining. You can also write expressions for a shape's dimensions in the **Link** editor. Below is an example of a doughnut shape (an elliptical ring sector) with an inner radius defined from the radius of a previous step's annotation.

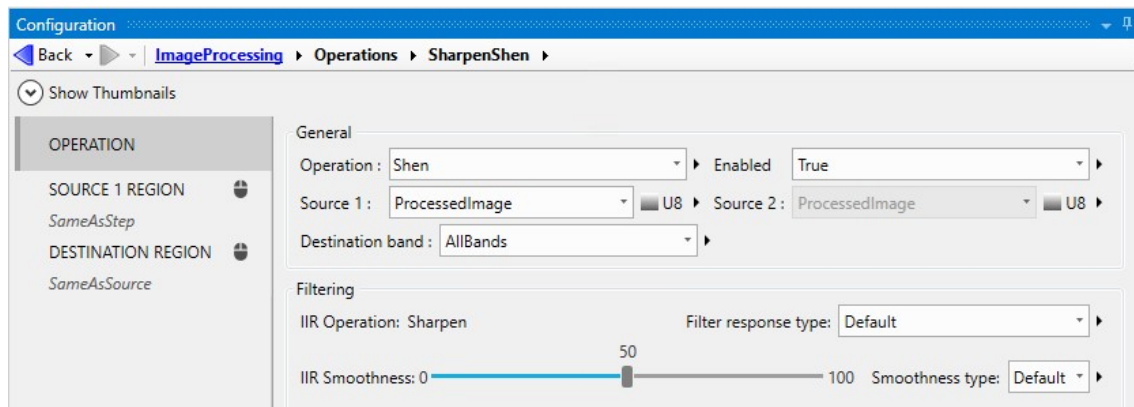


In some cases, when redefining a shape, if there are no expressions involved, it might be quicker to delete the current shape and draw a new one rather than editing its dimensions. To select the entire image as your search region from your current shape, click the **Use whole image** () toolbar button in the **Project** toolbar.

If a step's search region or test region is calculated through links or expressions, and the resulting search region falls entirely outside the image boundaries, an error will occur. This can happen when the calibration changes and throws the region out of range. You will be informed of this error during design-time.

▼ Active mouse button

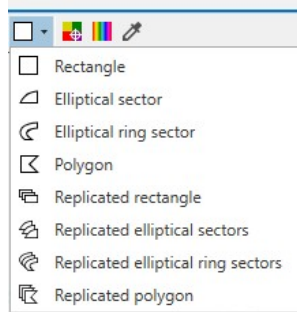
Some steps, like the **ImageProcessing** step, also allow you to define and redefine search regions using the **Active mouse** () button in the tab, shown in the image below:



▼ Replicated regions

Replicated regions, in analysis steps, and replicated annotation shapes both create a number of repeated shapes, each placed at a different position and orientation. The positions and orientations are specified as an array of fixtures or 3 arrays one each for X,Y and angle.

If a step supports replicated regions, then the **Define a region** () toolbar button will include them.



▼ Specifying the positions and orientations

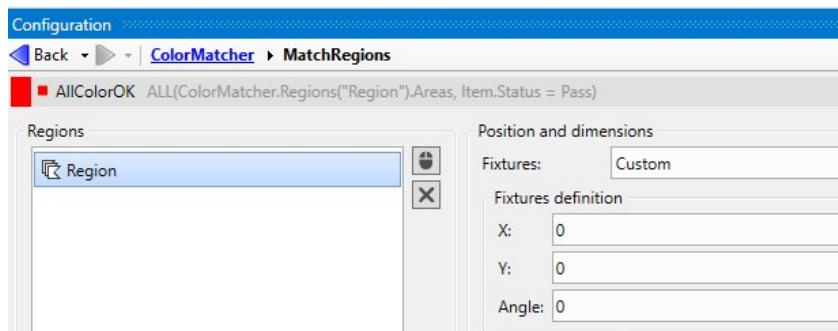
The most common way to specify where to replicate is to use the Fixture result of a step in the Locators group. To do so, you must create a [SELECT](#) expression in the Fixtures input. You can use the [Advanced](#) editor or copy and paste one of the following examples to build the [SELECT](#) expression for the locator step you are using.

- `SELECT(BlobAnalysis.BlobFound, Item.Fixture).`
- `SELECT(PatternMatching.Occurrences, Item.Fixture).`
- `SELECT(CircleFinder.Occurrences, Item.Fixture).`



It might be useful to replicate at each of a set of positions where the user has placed points in the operator view, or at each vertex of a polyline. For those cases where a Point Array already exists, you can use a [FIXTUREARRAY](#) function call in the form `FIXTUREARRAY(«SourceArrayPoint», «SourceArrayAngle»).`

Alternately, you can set Fixtures to Custom and provide individual x,y, and angle arrays. This is useful if you want to use the positions coming from a locator step but want to ignore or modify the angle. The Fixture result cannot be separated into its 3 components. If you just want to access the X-component and Y-component separately, then you need to use the Positions result. For example, to get the array of X components from all blobs use the expression `SELECT(BlobAnalysis.Positions.Points, Item.X).`



Dealing with color images

Matrox Design Assistant supports color acquisition. Color processing is supported with the **ColorMatcher** step and with certain operations in the **ImageProcessing** step.

Color images contain 3 bands of data. Images grabbed in color mode from a color camera have red, green, and blue bands. Note that many of the Matrox Design Assistant steps only accept 1-band images.

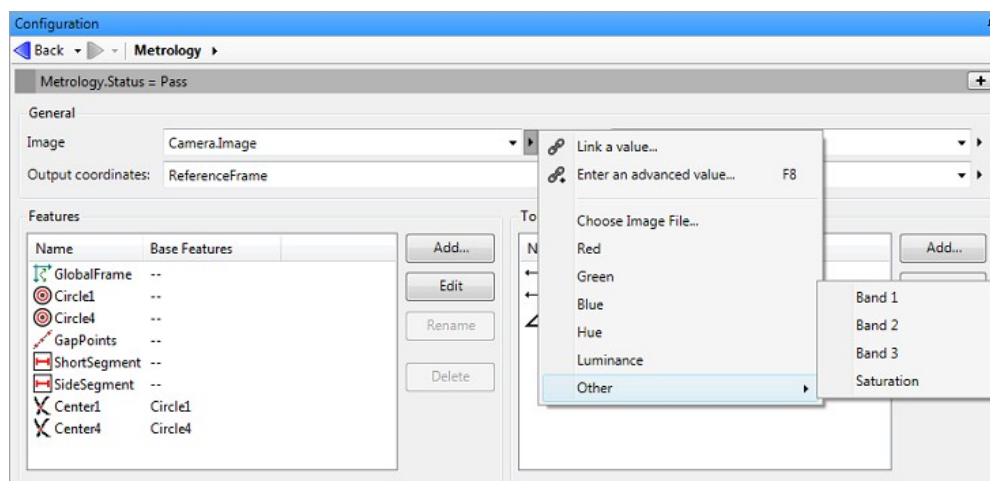
▼ Generally working with color

In some cases, it is more appropriate to convert the image to another format (for example, Hue (color information), Saturation (strength of the color), and Luminance) which is proportional to the intensity that would be captured with a monochrome camera. The **ImageProcessing** step has a **Convert** operation in the **Copy and Convert** category (along with **Fill**, **Copy**, and **CopyConditional**) that lets you convert an incoming RGB image into either a 3-band HSL image or into a 1-band image containing either hue, saturation, or luminance. On the first page of the **Configuration** pane of the **ImageProcessing** step you can specify if you want to produce a 1 or 3-band resulting image.

When a 3-band image is saved to a file, the color space type (RGB or HSL) is not saved in the file. By default, a 3-band image is considered to be RGB. If you have saved an HSL image and want to load it either through the **Camera** step, the **LoadImage** step, or through the static image repository, you must indicate that it is an HSL image using the **Type** property.

By default, when you add a step that only works with grayscale, the image will convert to grayscale using the image's luminance band. However, sometimes you might still receive a message indicating that the input image is not valid because the step only works on grayscale images. You will be prompted to select one band of your color input image, or to use an image processing step to convert the input image to grayscale. Perform one of the following options to clear this message.

- If the camera used to grab images supports both color and monochrome, you can choose to set the camera color mode to grayscale in the **Platform Configuration** dialog. You will need to re-grab your images so they appear in grayscale.
- If the project will not be using the color information, select the **Convert to grayscale** option in the **Image sets** page of the **Platform Configuration** dialog. By default, the **Camera** step will produce 1-band images (that is, grayscale images).
- You can choose to work on one band individually, by linking the analysis step's **Image** property to one of the bands specified in the **Alternate** menu (sideways triangle). For an RGB image, selecting **Red**, **Green**, or **Blue** will extract that band from the original image, while selecting **Hue** will perform a conversion to produce a color-dependent hue image. Selecting **Luminance** will provide a grayscale equivalent of the original image. Note that if you have already used image processing to convert to an HSL, 3-band image, HS and L are available. Selecting **Other** gives access to the less commonly used **Saturation** (which requires that you are working with an image that has already been converted to HSL using an explicit **ImageProcessing** step). **Other** also offers the generic **Band 1**, **Band 2**, and **Band 3** options.



- Use the **ImageProcessing** step to produce a Luminance image (RGB to L) that you can use for operations such as locating or matching.

In Matrox Design Assistant, color components in a **tree structure** can also be referenced by color band with an index value, using the **BandAtIndex** notation. This is useful if the index depends on elements such as a variable or a loop counter.

- Red: `Camera.Image.Bands.BandAtIndex(1)`
- Green: `Camera.Image.Bands.BandAtIndex(2)`

- **Blue:** `Camera.Image.Bands.BandAtIndex(3)`.

▼ Color matching

You can perform simple color determination (distinguishing between strong, uniform colors like red, green, blue, and yellow) in a number of ways:

- Convert your image to HSL and use a single **IntensityChecker** step on the hue band. This approach requires extra care when dealing with black or white objects because their hue is not defined. Similarly, red objects can have hue values in 2 different ranges (for example, 0 to 10 and 245 to 255). A sample project is provided as an example. See [Chapter 12: IntensityChecker step](#) for more information.
- Use multiple **IntensityChecker** steps on regions in each band, and compare 2 or 3 of the average values to expected ranges. For example, when distinguishing between red, green, and blue in a situation where there isn't enough time to convert to hue.
- Use the difference between 2 color bands to distinguish colors. Red, green, and yellow (for example, in LEDs, traffic lights, and signal lights) can be reliably distinguished using the resulting signed image buffer derived from the red image minus the green image. Red will give positive numbers, green will give negative, and yellow (which is equal parts red and green) will give values close to zero. See the [Arithmetic](#) subsection of the [ImageProcessing step operations](#) section in [Chapter 9: ImageProcessing step](#) for more information.
- Use white balancing to have a more accurate representation of the color temperature of an image. For more information, see [Chapter 35: White balance](#).

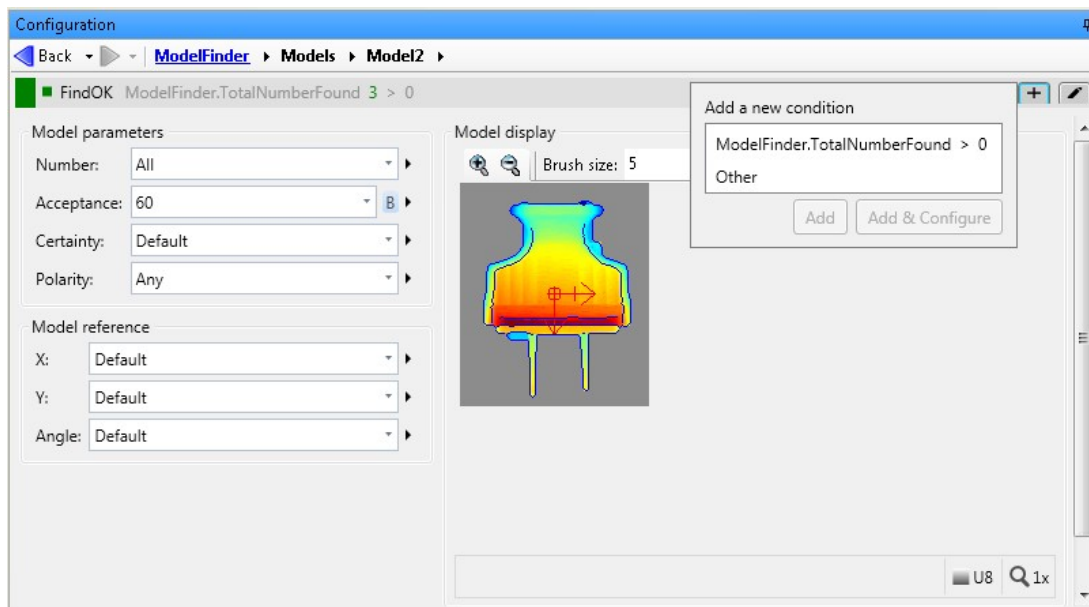
Note that for reliable color matching, the camera should be white balanced. For information on white balancing, refer to [Chapter 35: White balance](#).

If you require sophisticated color classification and matching, it might not be possible to distinguish between closely related colors with simple operations on the RGB and HSL images. To optimally convert a color image to grayscale, use the [Projection](#) operation from the **ImageProcessing** step. To perform [color matching](#), use the **ColorMatcher** step.

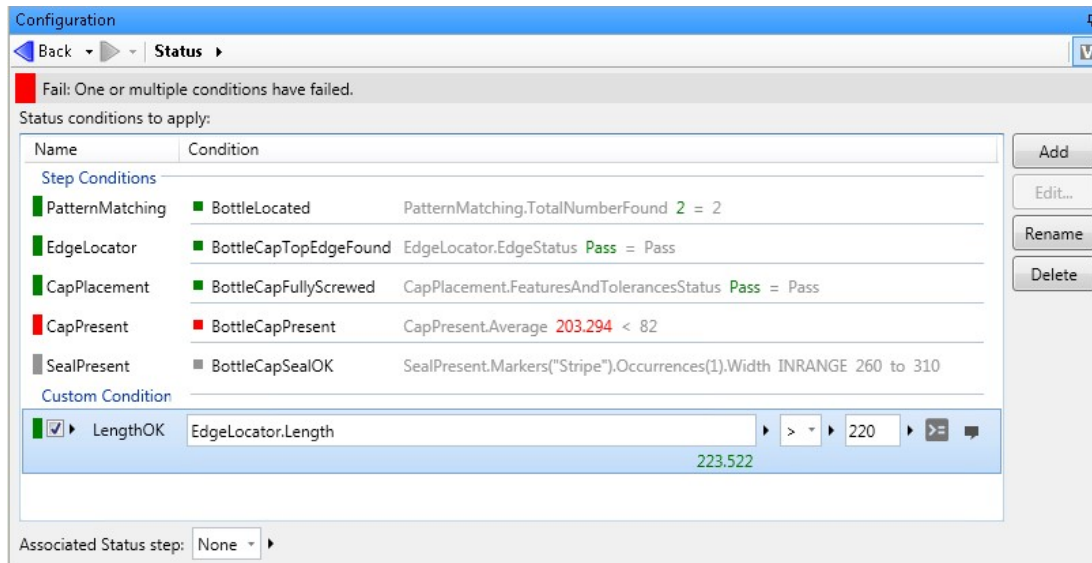
Status step

An important step in any project is the **Status** step. Among other uses, the **Status** step establishes the overall pass or fail state of your project, based on conditions that verify results from other steps.

Conditions that are associated with a specific step are referred to as [step conditions](#). For example, if you are using the **ModelFinder** step, a typical step condition would verify that you found a model occurrence. In analysis steps, a step condition is automatically generated when you create the step.



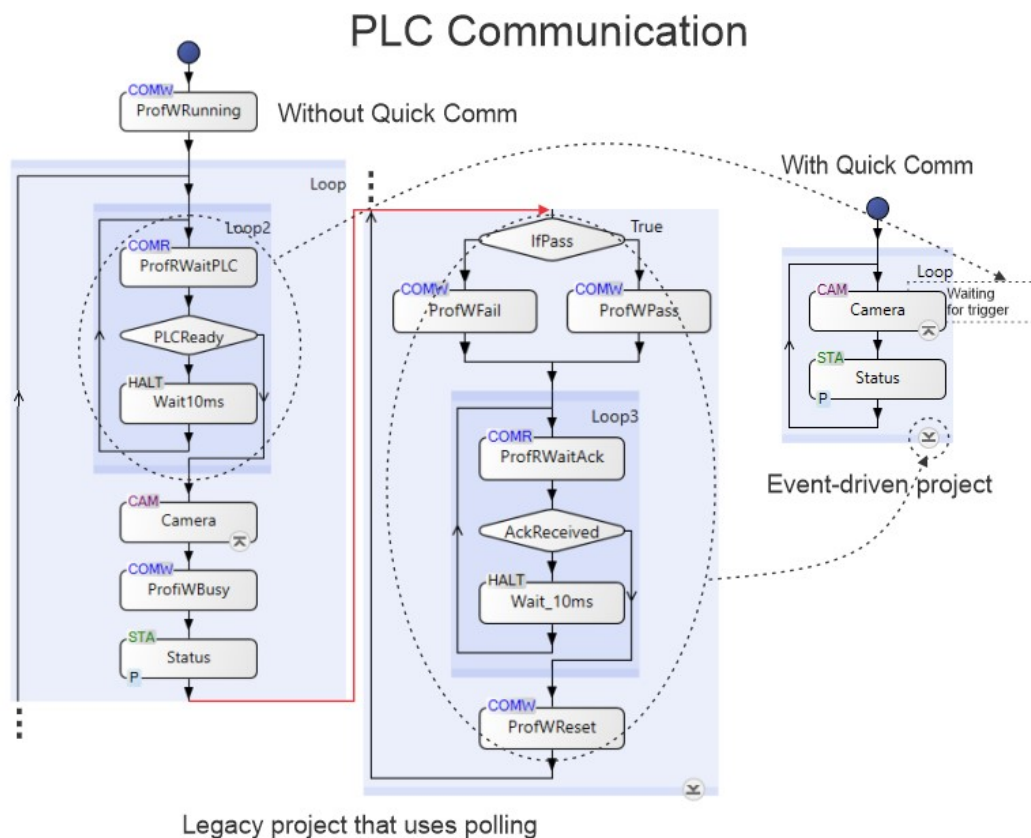
Custom conditions, which are not associated with any one step but rather with the **Status** step itself, also contribute to the **Status** step's overall pass/fail result. You can add a custom condition by clicking on the **Add** button in the **Configuration** pane of the **Status** step.



For more information, see the [Procedure for using the Status step and Error step](#) section in [Chapter 27: Flow control steps](#).

Events and actions

The recommended structure of a project is for the main flowchart to only contain the main inspection loop, which is the part that actively inspects images. You should place all other actions, such as configuration and recipe selection, in [subflowcharts](#) that will be called on operator view or PLC events. Common actions, such as [operator view publishing](#), [PLC handshakes](#), [output resets](#), [variable resets](#), and [internal annotation buffer clearing](#), can also be handled automatically as background activity not requiring flowchart steps. For instance, PLC communication actions are event-driven when Quick Comm is enabled. Such event-driven architecture means that the main flowchart need not continuously poll (check) for requests from the operator view or changes in incoming PLC data, as was the case for legacy projects. Event-driven projects can therefore be designed more efficiently.



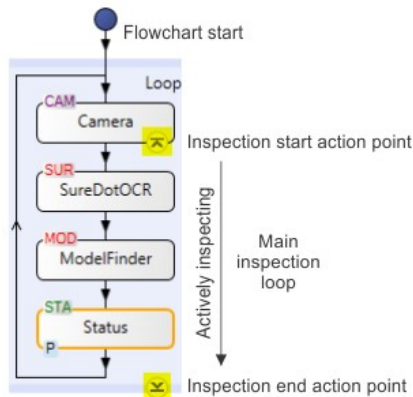
Note that legacy projects needed extra steps to poll for input changes. This is still the case for projects waiting for value changes on serial port, discrete I/O, Modbus, and TCP/IP communication. Although this is no longer required for EtherNet/IP or PROFINET, such projects will function as before.

▼ Action points

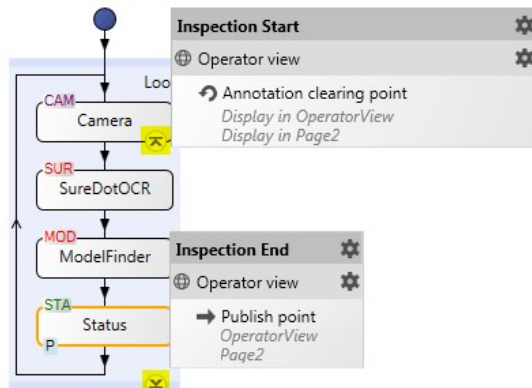
Action points refer to specific points in the flowchart, when certain actions occur (such as operator view updates, PLC communication, and internal annotation buffer clearing). The main action points are inspection start and inspection end; however, you can add additional action points (for example, based on PLC fields).

▼ Inspection start and inspection end (main inspection loop)

The inspection start and inspection end action points typically mark the beginning and end of the main inspection. During this time, the project is actively inspecting (analyzing images), and in its main inspection loop.



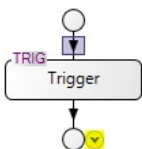
By default, the moment when the main flowchart fetches a new image is considered to be the inspection start action point, and the bottom of the main inspection loop is considered the inspection end action point. These 2 action points are marked on the main flowchart with circular arrowhead symbols showing the top (⌂) of the inspection, and the bottom (⌂) of the inspection. Clicking on one of these buttons lists the actions that will take place when the point is reached. For information about actions that can happen at the inspection end point when using Quick Comm, see the [Status and control handshake](#) subsection of the [Quick Comm handshake](#) section in [Chapter 53: PROFINET, EtherNet/IP, CC-Link, and Quick Comm](#).



▼ Adding other action points

Flowcharts (including subflowcharts) can also have action points that are not inspection start or end. Such action points are indicated by similar arrowhead symbols and typically occur at the start (arrowhead pointing up) or end (arrowhead pointing down) of a step or subflowchart. A typical reason to add an action point is to explicitly update information in the operator view at the end of a subflowchart (by default, operator view publishing happens at the inspection end action point). To add such a point, use the **Operator View Publishing Synchronization** dialog, accessible from the **Operator Views Synchronize publishing...** menu item.

You can also define an action point to add an additional PLC handshake or send data. To add such a point, use the **Actions** page in the **Platform Configuration** dialog. Clicking the **Add Action** button allows you to place the action in the flowchart, (for example, OnPLCChanged (PassFailConditions) in the Matrox Design Assistant CodeReader template project).



To see the actions that will happen when the flowchart reaches an action point, click on the action point symbol.

Note that advanced users can add and modify action points that interact with industrial protocol data using the **Actions** page of the **Industrial Protocols** page in the **Platform**


Configuration dialog. For an overview diagram of events and actions, see [Configuring events and PLC handshake actions](#) .

▼ When actions occur

Many types of actions occur during the course of a running project. Some actions are handled immediately, some are scheduled (synchronous), and some happen at unknown moments (asynchronous).

Acquisition trigger actions, along with any reset action triggered by a PLC reset request in the Quick Comm control section, are handled immediately. They are independent of action points.

Certain routine actions, such as updating the operator view (publishing) and Quick Comm handshaking, are scheduled to happen when the flowchart reaches their specified action points. Such actions always occur at the same points in the flowchart; they are called synchronous.

Changes in an [operator view input element](#) (such as pressing a command button or selecting a new value from a dropdown list), or events tied to [PLC field changes](#) are asynchronous events, and can be configured to execute a subflowchart. Such events typically occur at unknown moments. These events cannot be handled while a project is between inspection start and inspection end (actively inspecting), and are queued until the inspection end action point. At design-time, the **Events pending** () toolbar button in the **Platform** toolbar, shows the number of events that are queued.

▼ Operator view events

You can configure an operator view element (such as a command button, dropdown list, and text input box) to generate an event when its value changes. This executes a subflowchart. For more information, see the [Configuring input elements](#) section in [Chapter 63: Customizing the operator view](#).

Some typical scenarios of when you would execute a subflowchart from the operator view are:

- To redefine a new match model, using the [Reconfigure](#) step.
- To switch recipes, using the [LoadRecipe](#) step.
- To train a new recipe, using the [CreateRecipe](#) step.
- To instantly view changes made through the operator view to the current image, using the [Trigger](#) step. For instance, this is done by calling the OnSendTrigger subflowchart in Matrox Design Assistant template projects.

All 4 of these techniques are used in template projects installed with Matrox Design Assistant.

It is generally not logical to asynchronously change settings partway through inspection. Therefore, if a project is in its [main inspection loop](#) when the event happens, the associated action (for example, executing a subflowchart) will not run immediately; it will be added to a queue of pending actions. Any pending requests to run a subflowchart are handled in the sequence they arrived, at the inspection end action point. Events that happen outside the main inspection loop are handled immediately.

When a flowchart is waiting for a trigger at a [Camera](#) step, and there is a request to execute a subflowchart (from the operator view or PLC), the wait for that trigger is suspended, and the requested subflowchart is run; then, the [Camera](#) step resumes. Details about this process can be seen in the Matrox Design Assistant template projects that use a [software trigger](#) mode for their [operator view pages](#) associated with training (models).

Note that if a running project is waiting for an event, there is no visible indicator from the operator view, unless the [wait banner](#) is enabled.

For information on configuring PLC inputs to generate subflowchart execution events, see the [Subflowchart execution events](#) subsection of the [Configuring PLC handshake actions and subflowchart execution events](#) section in [Chapter 53: PROFINET, EtherNet/IP, CC-Link, and Quick Comm](#).

▼ Types of actions

There are 3 types of event-driven actions:

- [Operator view publishing](#).
- [PLC handshake](#).
- [Subflowchart execution](#).

For information about viewing a list of these actions, or executing these actions in your project at design-time, see the [Testing event-driven actions](#) subsection of the [Testing and debugging a project at design-time](#) section later in this chapter.

▼ Operator view publishing

The operator view publishing actions are [publish](#) and [clear](#). They are present in all projects.

By default, publishing is performed at the inspection end action point and annotations are cleared at the inspection start action point. However, you can set them to occur at different action points. For information about managing publishing groups and changing when they happen, see the [Controlling when data is sent to the operator view](#) section in [Chapter 63: Customizing the operator view](#).

▼ PLC handshake

The PLC handshake actions are [set](#), [wait](#), and [clear](#).



The events that fire (trigger) these actions occur when the flowchart reaches an action point or when the PLC sets certain control bits.

The set, wait, and clear actions are a key part of [Quick Comm](#). The basic Quick Comm handshake uses 6 bits, and optionally 1 trigger bit, per camera. You can also add **Auto DataToPLC** fields, from the [Quick Watch](#) flyout panel or from the [Platform Configuration](#) dialog; this will transfer the linked values of these fields at the inspection end action point. By default, a [wait banner](#) appears in the operator view if a running project is waiting for a handshake from the PLC.

For information about how to reorder and set data types for the fields, see [Chapter 53: PROFINET, EtherNet/IP, CC-Link, and Quick Comm](#). For information about triggers and timers, see the [Triggering](#) section in [Chapter 32: Acquisition](#) and also see [Chapter 52: IO steps](#).

▼ Subflowchart execution

Subflowchart execution actions run a subflowchart when the project is outside of the main inspection loop. The events that can fire (execute) these actions are:

- Operator view button clicks.
- Operator view data that was changed in an input element.
- Changes in data from PLC [data tables](#).

Additional Quick Comm bit fields have been predefined for communicating common requests like reset or recipe changes. 2 general control soft event bit fields and 2 soft event bit fields per camera module have also been defined; for example, the **PLCSoftEvent1** field. For more information about how to add an event and associate a flowchart with any of the control bit fields set by the PLC, see [Chapter 53: PROFINET, EtherNet/IP, CC-Link, and Quick Comm](#).

Specifying reset points for variables and step outputs

You can specify the reset points for variables and step outputs, using any of the settings available in Matrox Design Assistant.

▼ Reset points for variables

When you create a variable, you can specify its **ResetPoint** in the [Add variable](#) dialog. You can modify reset points using the [Edit variables](#) dialog.

You can set the **ResetPoint** for variables to **InspectionStart**, **NeverReset**, or **FlowchartStart**, respectively. **InspectionStart** specifies that the variables will be reset to their initial value at the project's [inspection start](#) action point. To never reset variables, or to only reset them once when the project starts, set **ResetPoint** to **NeverReset** or **FlowchartStart**, respectively. For more information, see the [Lifetime](#) section in [Chapter 29: Variables and the Store step](#).

▼ Reset points for step outputs

Steps between the main flowchart's [inspection start](#) (⏏) and [inspection end](#) (⏏), have a **ResetPoint** property that establishes the action point at which to reset the outputs of the steps. Steps in subflowcharts can only be reset if their subflowchart is called by the main flowchart. That is, steps in [event subflowcharts](#) do not have the **ResetPoint** property; they keep their values for the duration of the project. However, they are not [persistent](#).

The **ResetPoint** property of a step is set from the step's [Properties](#) pane. You can set the **ResetPoint** property for a step's outputs to **InspectionStart** (the default setting) or **NeverReset**.

InspectionStart specifies that the step's outputs get cleared for each new inspection loop. **NeverReset** specifies that the step remembers its previous results; this can be useful when the flowchart has several branches, and the results of a step must be made available to subsequent loops. Be aware that not resetting a step's outputs makes them available forever, which can cause stale data in the [operator view](#).

Note that if you select **InspectionStart**, a step's outputs have no values after inspection start, and before the step is run. As a result, [linking](#) to the step's outputs can display an error in the [Quick Evaluate](#) pane. For instance, in the Matrox Design Assistant **BottleInspection** example project, linking to an output of the **SealPresent** step could display the error: Retrieving the 'Occurrences' output from 'SealPresent.Markers("Stripe")' has failed. The **SealPresent** step has not run yet or was reset at inspection start.

This error is avoided if the step's outputs are validated before use. Use the [CHECKVALID](#) and [ISVALID](#) functions to ensure expressions that link to results do not cause errors.

You can use reset points to ensure that the results sent to the PLC (or operator view) are relevant to the current loop, and not leftover from previous loops.

Inspection labels

Inspection labels are color coded tags applied to images in the [Camera Images](#) pane or the Filmstrip element in the operator view. Each inspection label has a name, a configurable foreground color, a predetermined background color, a value, a condition, and a visibility. When an inspection label is applied to an image, the name and value of the inspection label appear.

An inspection label's value is the result of an expression. For example, if an inspection label has its value set to `BlobAnalysis.NumberOfBlobsFound`, then when the inspection label appears, it displays the number of blobs found. An expression label can be visible based on a condition that evaluates to true or false, such as `BlobAnalysis.NumberOfBlobsFound < 5`. When an inspection label's condition is true, the inspection label is applied to the image. An inspection label's visibility determines whether the inspection label will be visible on the images in the [Camera Images](#) pane (design-time), the Filmstrip element in the operator view (runtime), or both.

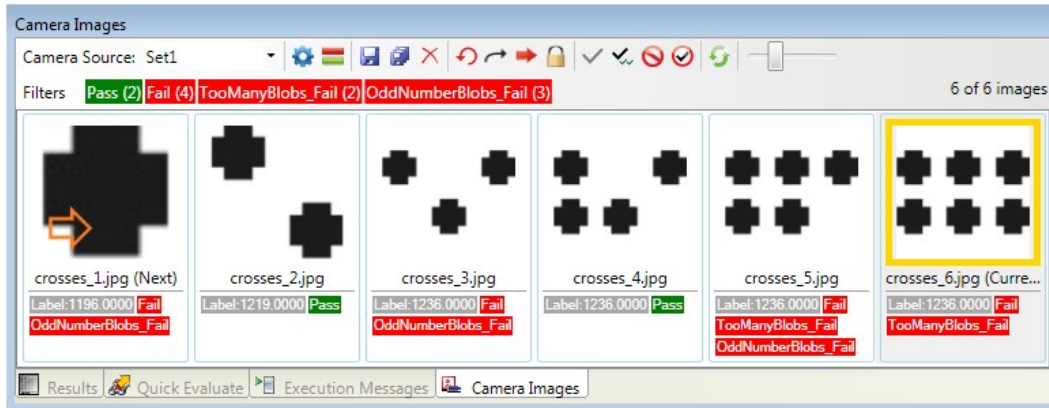
By default, there is an automatically added label for each condition in the default **Status** step in the main flowchart. To disable the generation of an automatic label for a condition, click on the step associated with that condition, navigate to the [Properties](#) pane, and set **CreateAutomaticLabels** to false.

▼ Types of inspection labels

There are 3 categories of inspection labels: data, pass, and fail. By default, these are set to a background color of gray, green, and red, respectively. The data inspection labels display some information, such as a time stamp or loop count. A pass inspection label displays when an image passes its **Status** step conditions and a fail inspection label displays when an image fails its **Status** step conditions. Typically there will only be a single pass inspection label, but several fail inspection labels, one for every condition that fails. The inspection labels can be quickly identified by their color.

By default, a project starts with 4 inspection labels: T, Pass, Fail, and ErrorCaught. A T label displays a gray timestamp, Pass and Fail labels are discussed above, and an ErrorCaught label signifies that the **Status** step caught an error from either an **Error** step or another source. For more information, see the [Catching and throwing exceptions](#) subsection of the [Procedure for using the Status step and Error step](#) section in [Chapter 27: Flow control steps](#).

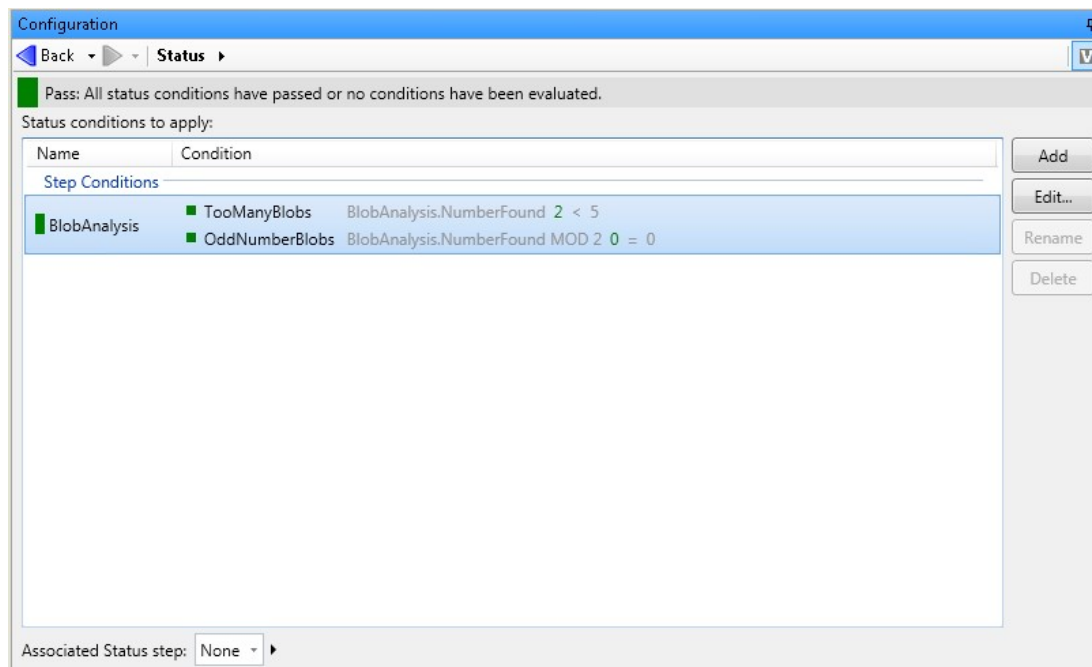
The following example shows a **Camera Images** pane with images and several inspection labels. The gray data inspection label is on all images and displays the size of the first blob found. The red inspection label called **OddNumberBlobs_Fail** is on the images that failed because of an odd number of blobs. The red inspection label called **TooManyBlobs_Fail** is on the images that failed because they have 5 or more blobs (note that after the inspection label name, the number of blobs in the image appears). The global inspection label called **Fail** is on the images that have one or more fail inspection labels. The green inspection label called **Pass** is on the images that passed all conditions in the **Status** step.



▼ Creating new inspection labels

There are multiple ways to create a new inspection label:

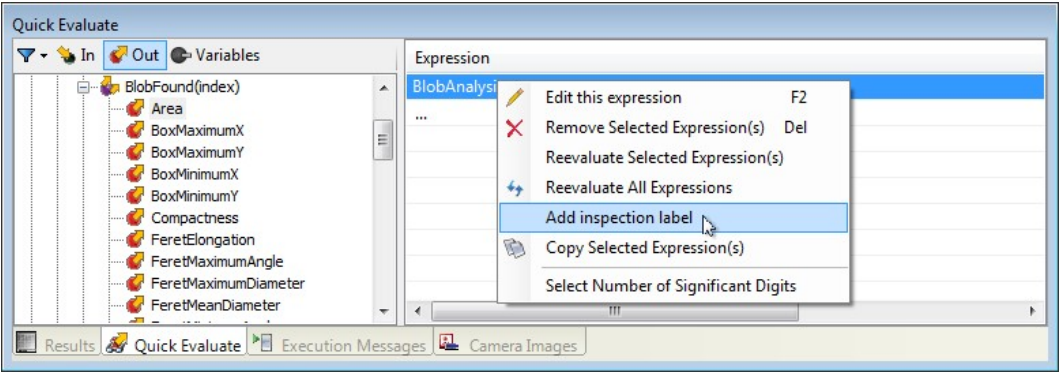
- Add a new condition to a **Status** step. This is the most common way to add an inspection label and you should use it whenever possible. This will create an inspection label with a Fail condition in the **Manage Inspection Labels** dialog (further explained below), and the value field will be left blank. If you have more than one **Status** step in your project, use the **Automatic Labels...** button in the **Manage Inspection Labels** dialog to select which **Status** step will generate automatic labels. See the [Procedure for using the Status step and Error step](#) section in [Chapter 27: Flow control steps](#) for more information. The image below demonstrates some conditions that have been added in the **Status** step and the results in labels from the previous image.



Note that if a **Status** step is part of a recipe, it cannot automatically generate labels.

- Select an expression in the **Quick Evaluate** pane, right-click it, and select the **Add inspection label** menu item. The **Manage Inspection Labels** dialog will appear, allowing

you to modify your new inspection label. The new inspection label's value will be that of the expression selected and will have no condition. You might want to use this method if you want to display additional data to your images in the **Camera Images** pane. The image below demonstrates how to add an inspection label from the **Quick Evaluate** pane.



Note that you can easily add relevant expressions using the **Quick Watch** flyout panel. You can right-click data in the panel and select **Add to Quick Evaluate** to add them to the **Quick Evaluate** pane.

- Click on **Add** in the **Manage Inspection Labels** dialog.

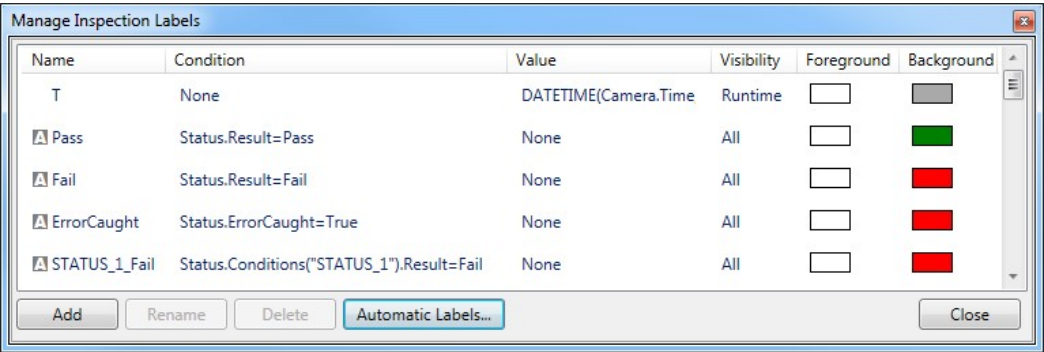
▼ **Manage Inspection Labels dialog**

To edit inspection labels, use the **Manage Inspection Labels** dialog, accessible from the **Project** menu or the toolbar of the **Camera Images** pane. You can also create new inspection labels from this dialog.

Edit an inspection label by selecting it in the dialog and clicking on the item to edit. Regardless of the category of inspection label, the default foreground color, used for the font, is white, and the visibility, used to determine where the inspection label is visible, is set to Both. Typically, you will add a condition for a data inspection label, and/or a value for a fail inspection label. You can also set the background color to create meaningful color-coded categories of inspection labels.

When adding an inspection label using the **Manage Inspection Labels** dialog, the default background color is gray, and the Value and Condition fields are empty. You must fill these fields manually, usually by copying and pasting from the **Advanced** editor of the **Configuration** pane of another step.

In the figure below, you see the **Manage Inspection Labels** dialog with the default inspection labels and the user-defined inspection labels. Note that when the Condition field is empty, the inspection label is always visible, but will still only appear in the locations specified in the Visibility field.



You can click on the **Automatic Labels...** button to have **Status steps** that are not part of a **recipe** create inspection labels automatically. Note that if the property **CreateAutomaticLabels** is set to false, the label will not be created. You can see which labels have been added automatically by the symbol added to the left of their name.

Communication

Your runtime platform is capable of communicating with external devices using a number of different protocols and hardware connections. For most configurations, Matrox Design Assistant has a series of steps that control the information flow. See the communication methods and steps from the table below for more information. The **OPC UA** protocol does not rely on steps for reading and writing data, the protocol is bound directly to the inputs and outputs of steps that provide or received the data, or to variables. It is the only protocol that can transfer image data, through the use of image variables.

Communication method	Steps	Description
I/O	IOWriter step, IOWriter step	The IO steps allow you to send a signal to or receive a signal from an external hardware device, respectively, by using user-defined pins on your runtime platform's input and output connector. Platforms that have a Matrox Advanced I/O Engine have additional I/O capabilities.
Modbus	ModbusReader step, ModbusWriter step	The Modbus PLC steps allow you to communicate information, using the Modbus communication protocol, between your runtime platform and other Modbus devices, such as a programmable logic controller, sensor, or robot controller.
	CommReader step, CommWriter	The PLC steps allow you to transmit data between assemblies located on your runtime platform and another device

EtherNet/IP	step	(scanner) using the EtherNet/IP protocol. A scanner is typically a programmable logic controller (PLC). The Quick Comm feature provides a predefined event-driven handshake protocol.
PROFINET	CommReader step, CommWriter step	The PLC steps allow you to transmit data between your runtime platform and a controller using the PROFINET protocol. The Quick Comm feature provides a predefined event-driven handshake protocol. To use the PROFINET industrial communication protocol, you must be using a Matrox runtime platform (for example, Matrox Iris GTR/GTX or Matrox 4Sight EV6) or your runtime platform must have a hardware component with a Matrox PROFINET Engine (for example, Matrox Indio).
CC-Link IE Field Basic	CommReader step, CommWriter step	The PLC steps allow you to transmit data between link devices located on your runtime platform and another station using the CC-Link IE Field Basic protocol. The Quick Comm feature provides a predefined event-driven handshake protocol.
Network	NetworkConnection step, NetworkReader step, NetworkWriter step	The Network steps allow you to establish, validate, and terminate a connection with, as well as send data to and receive data from, another network client (such as a computer or network-aware device) on your network. There are 2 available networking protocols: UDP (User Datagram Protocol) and TCP (Transfer Control Protocol).
Serial port	SerialPortSetup step, SerialPortReader step, SerialPortWriter step	The Serial Port steps allow your runtime platform to exchange information with external devices, such as motion or light controllers, that are connected via a serial port (if available on your runtime platform).
Robot	RobotWriter step, RobotWait step, RobotParameters step	The Robot steps allow your runtime platform to communicate with connected robot controllers, typically to send position data to the robot controller.
OPC UA	Direct binding	Matrox Design Assistant implements an OPC Unified Architecture (OPC UA) server. It provides bindings, events and flowcharts, allowing the client to perform a wide range of activities, from simply monitoring the project, to efficiently receiving result data at the end of each inspection, through providing inputs through custom HMIs that have the same capabilities as our Operator Views.

Testing and debugging a project at design-time

Testing and debugging a project is often done hand in hand with developing that project. You will typically run, test, debug, and modify your project at design-time until it works properly. This section lists some tips to help testing and debugging projects at design-time, and also describes how to test event-driven actions.


For deployed projects at runtime, you can analyze the project's execution for debugging or optimization using the **MONITORING** portal page in the **Matrox Design Assistant configuration** portal. You can view details about each flowchart step that has run and its time of execution, as well as view statistics for the project such as average inspection times. See [Using the monitoring tab](#) for more information. Alternatively, you can generate a trace file for projects at runtime using [Matrox Profiler](#).

For information on the tools that Matrox Design Assistant offers to help you run your flowchart and access its results, see the [Techniques for running the flowchart and accessing results](#) section earlier in this chapter.

▼ Testing and debugging tips

The following is a brief list of tips and hints for testing and debugging your project at design-time:

- **Use the debug toolbar buttons.**

To perform general testing actions, use the [Debug toolbar buttons](#), accessible from the **Project** toolbar. For instance, to walk through your project a step at a time, and update the display and/or your results, use the **Next step** () toolbar button. You can also use the corresponding shortcut keys for each toolbar button, which can be found in the **Debug** menu. For example, you can run the loop to the next image and stop at the next step by pressing **F6**.

- **Use the Errors pane.**


To see any errors or warnings related to your Matrox Design Assistant project, use the **Errors** pane. You can double-click on an element in the tree to go directly to the location of the error or warning in the project.

- **Search with Edit Find.**

To quickly locate expressions, steps and operator view elements, or variables in your project, use the **Find** pane, accessible from the **Edit Find** menu item. You can double-click on an item in the **Find** pane to go to the corresponding step or element.

- **Run only required steps.**

- **Skip communication steps.**

If a project is running on your runtime platform while you are debugging another project on your development computer, you are not able to use any of the platform's communication hardware. To skip all communication steps in your flowchart, select the **Communications** () toolbar button in the **Platform** toolbar.

- **Quick Run mode.**

To quickly isolate and test small parts of large or complex projects (including legacy projects), use [Quick Run mode](#). Only the selected step(s), and the steps it depends on are run.

- **Disable steps.**

To make the flowchart skip one or more steps when it is run, select and then right-click on the step(s), and then select the **Disable selected step(s)** context menu item.

- **Select a step from the Step Log pane.**

To select a step in your project, click on the step in either the flowchart (ensure that you have selected the **Flowchart** tab) or the **Step Log** pane. Note that the **Step Log** pane only lists the steps that have run.

- **Use the Quick Evaluate pane.**

To display the current value of any step's input or output/result, as well as any expression, variable, or platform configuration setting, use the **Quick Evaluate** pane.

You can send links to results directly to the **Quick Evaluate** pane using the right mouse menu in the **Quick Watch** flyout panel. Alternatively, you can drag directly from the [tree structure](#) or from the expression column and drop into any input box or **Advanced** editor.

You can also use the **Quick Evaluate** pane as a temporary holding place to transfer links elsewhere. This is particularly useful for transferring results from **Flowchart step** images to **Value element** links in the [Operator View Layout](#) tab.

- **Choose images.**

- **Use image sets.**

If a project is running on your runtime platform while you are debugging another project on your development computer, you cannot access live grab images; therefore you

must work with [image sets](#).

o **Select the next image.**

To view, exclude, or select the next image to be used at design-time, use the file thumbnails in the [Camera Images](#) pane.

o **Lock the image.**

To stop your project from advancing through the sequence of images with which you are testing, **lock** the image in the [Camera Images](#) pane. This allows you to perform further validation on the configuration of your steps against particularly troublesome images.

o **Select image to display.**

For steps that do not normally have an image, you can select one from the **View Display Image** menu item.

For information about testing your project's operator view at design-time, see the [Testing and debugging operator views at design-time](#) section in [Chapter 63: Customizing the operator view](#).

▼ **Testing PLC interaction**


You can test the communication steps in your project in different ways, depending on whether your runtime platform is connected to a PLC during development.

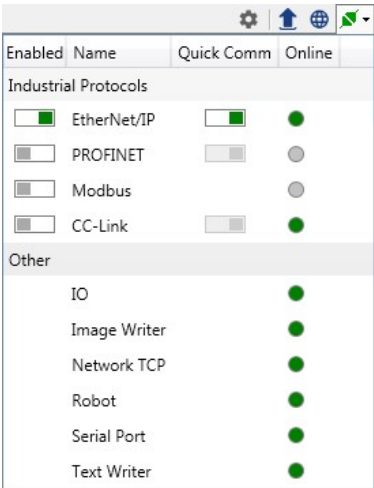
When you run your project in emulation mode, all communication and file reader/writer steps are skipped. You therefore cannot test PLC interaction in emulation mode.

If your development computer is connected to a runtime platform that has the appropriate communication hardware and is connected to a PLC, you can test your project by configuring your PLC to receive data and send the responses it will send in the production environment. You can see the values of the data fields and IO bits on the relevant tab of the [COMMS](#) portal page of the [Matrox Design Assistant configuration](#) portal.

If your development computer is connected to a runtime platform that is not connected to a PLC and you are using Quick Comm, you can enable PLC emulation for the industrial protocol you are using in the relevant tab of the [COMMS](#) portal page of the [Matrox Design Assistant configuration](#) portal. For more information, see [PLC emulation](#).

For the [MODBUS](#) protocol third party applications such as ModBusPoll and ModBusMaster can stand in for the actual PLC or Device hardware.

You can temporarily enable or disable some or all steps that perform communication while testing your project. To do so, access the [Communications](#) dropdown panel by selecting the [Communications](#)  toolbar button in the **Platform** toolbar.

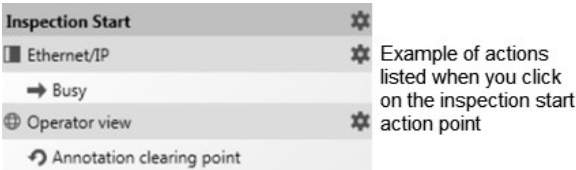


▼ **Testing event-driven actions**

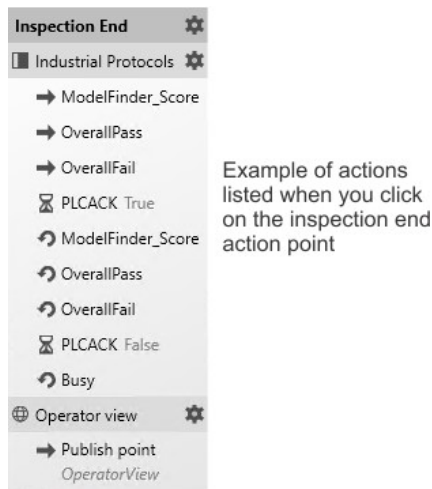
As previously discussed in [Events and actions](#), there are several kinds of event-driven actions (operator view publishing, PLC handshake, and subflowchart execution). While testing your project at design-time, it can be helpful to view or execute these actions in that project.

▼ **Viewing action lists**

To view the actions that happen when the flowchart reaches an action point, click on the action point, such as inspection start or inspection end.



The number of actions listed depend on whether you have enabled Quick Comm, which automatically configures a handshake with the PLC, and whether you have created any [Auto DataToPLC fields](#).

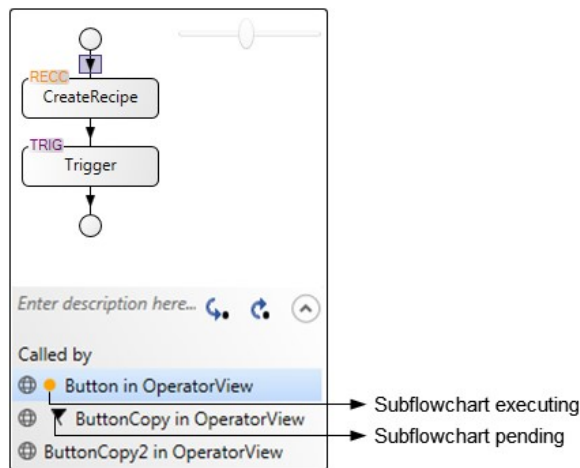


Example of actions listed when you click on the inspection end action point

Clicking on any of the gear symbols, at the top-right of the listed actions, takes you to an advanced settings dialog where you can make modifications to the corresponding action. Click on the **Actions** tab to enable, disable, or change an action.

▼ Viewing currently executing actions or pending actions

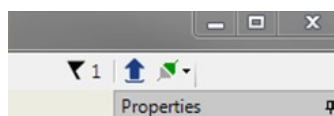
While viewing a subflowchart that is set to run upon an event, the bottom of the flowchart pane displays an orange dot to indicate that the subflowchart is currently executing due to an event. The flag indicates that a call to the corresponding flowchart is pending.



The circular symbol (shown below) indicates that the subflowchart is configured to be called by a subflowchart execution event from the operator view, while the rectangular symbol (shown below) indicates that it is also called from a **Flowchart step**.







When accessing the **Operator View Layout** tab, you can [simulate a button click](#) from the **Interface Actions** pane. After doing so, a flag in the **Platform** toolbar (the **Events pending** toolbar button) can appear with a number, to indicate how many subflowcharts are pending.



Simulating a button click lets you mimic a user clicking the button that executes the subflowchart from the operator view.

▼ Forcing the execution of a subflowchart that runs upon an event

To test (debug) a subflowchart that is typically called by an event at runtime, [access the subflowchart](#) and click the **Step in** () button. After [testing the subflowchart](#), click on the **Step out** () button, otherwise you can experience difficulties with other parts of the project at design-time.

Viewing the subflowcharts listed under **Called by** and using the **Step in** () button and **Step out** () button can be considered ways to navigate your subflowcharts.

Note that projects can also have [subflowcharts that are not called by an event](#). These subflowcharts are executed, like other steps, when [testing the project at design-time](#).

Setting up the operator view

After testing and error checking your project, you might want to design and edit the operator view, by accessing the **Operator View Layout** tab. The operator view is your deployed project's interface.

At design-time, neither inputs (images and information) nor results are displayed in the **Operator View Layout** tab. At runtime, the operator view is published as a web page that can have constantly updated images, results, user-entered data, and notifications. You must use elements to show information in, and accept information from, the operator view. For example, a Display element can show the image being grabbed, while a TextBox element can accept a value, such as specifying the number of models to find.

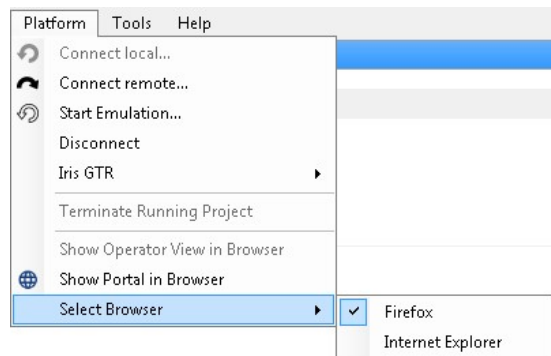
The operator view uses JavaScript to exchange images, results, inputs, and execution messages with the project deployed on the runtime platform. You can control when the project data is sent or published to the operator view using the **Operator View Publishing Synchronization** dialog.

For more information, see [Chapter 63: Customizing the operator view](#).

Deploying and running your project

When you deploy a project, the executable version of that project is copied to your runtime platform. Once deployed, the project is run autonomously on your platform. You can still use Matrox Design Assistant to interact with the project in a limited fashion, but you can also safely close Matrox Design Assistant. You can run and manage any project that has already been deployed to a platform from the **Matrox Design Assistant configuration** portal. For information on platforms, see the [Connecting to a runtime platform](#) section earlier in this chapter.

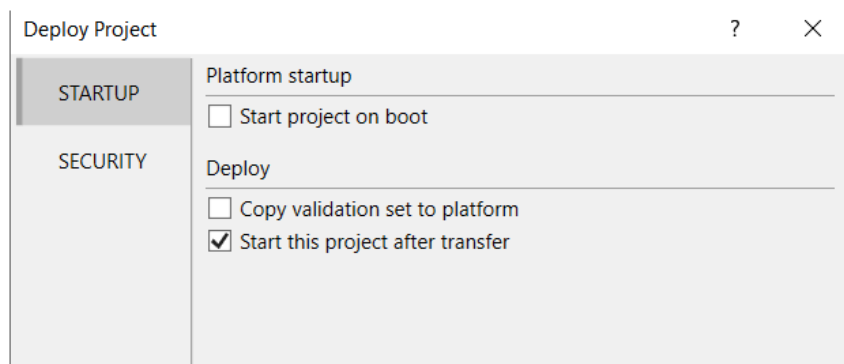
To select your web browser of choice, open the **Platform** menu, which is found in the main menu, and navigate to **Select Browser**.



▼ Deploy to platform

To deploy your project, perform the following:

1. Select the **Platform Deploy project** menu item or click on the **Deploy project** () toolbar button in the **Platform** toolbar. The **Deploy Project** dialog appears.



- Optionally, so that the project runs immediately once it is deployed, leave the **Start this project after transfer** option enabled.
- Optionally, copy the [validation set](#) to the platform. This is useful if you renamed a project with Project **SaveAs**, but you want the new project to start with the same validation set as the original.

Note: the validation set is copied from a runtime platform to the design time when you do **File Import Project**.

- Optionally, set the project to start automatically every time your platform is initialized (powered up or rebooted). This setting can be changed later in the **STARTUP** tab of the **PROJECTS** portal page of the **Matrox Design Assistant configuration** portal. You can enable this setting for multiple projects, if you want to start and run them all simultaneously on the runtime platform (provided your runtime platform has sufficient resources). For more information, see the [Considerations when running multiple projects simultaneously](#) section in [Chapter 72: Running multiple projects on a runtime platform](#).
- Optionally, if the project is set to start automatically every time the platform is initialized, and if you have a display screen directly connected to your runtime platform, you can enable the **Start the operator view automatically when the project starts** option.
On a PC platform, if the PC is not configured to automatically login before the project starts, the user will be prompted to enter a user name and password. Use a third-party program to allow automatic logon (for example, Autologon, which you can find at <http://technet.microsoft.com>) or enable it through the **AUTO LOGON** tab in the **DEVICE** portal page if you are working with a Matrox Iris GTR/GTX smart camera and want to show the operator view at boot. You do not need to enable autologon at startup to have the runtime start.
- Optionally, if the operator view is set to open automatically, you can choose to have it open in [Kiosk mode](#).
- Optionally, you can lock the project. For information, see the [Encrypting and locking a project](#) section in [Chapter 73: Security](#).
- Click on the **Deploy** button. The project will be deployed. Note that deployment might take some time. Even when the project starts running, there might be up to a minute delay before the operator view web page is updated and available.

If industrial communication protocols are enabled (for example, PROFINET), and one or more communication protocols are offline, you will get a warning. For more information, see the [Going offline](#) subsection of the [Communications dropdown panel](#) section in [Chapter 51: Communication overview](#).

If deployment is successful, and you have selected to run the project immediately, your project's operator view web page will open in your web browser and a message about the successful start of your project will appear in the [Execution Messages](#) pane.

▼ Files downloaded during deployment

During deployment, the project is saved and the flowchart files, static images, custom steps, and any MIL context files are downloaded to the platform for all recipes that have been selected for deployment. The operator view files are merged with template files containing the necessary scripts to let the operator view exchange information with the runtime project executable, and the resulting html file is sent to the platform. For more information about the operator view files, see [Chapter 63: Customizing the operator view](#).

▼ Recipes and persistent data

During deployment you must select which recipes will be downloaded to your runtime platform. The first time you deploy your project to a runtime platform, all recipes will be deployed by default. If you re-deploy your project to the same runtime platform, only the current recipe will be deployed by default. In either case, you can change which recipes will be deployed from the **Deploy Project** dialog. For more information about recipes, see the [Recipes overview](#) section in [Chapter 67: Recipes](#).

All steps which change per-recipe are persistent, as are many other elements of your project. When you re-deploy your project, all persistent elements on the runtime platform will have their values replaced by the values you have set on your development computer. If you need to retain the persistent data that is currently on your runtime platform, such as changes to a recipe made from the operator view, use the synchronization buttons in the [Recipes](#) pane. For more information about persistence, see the [Persistence and the SavePersistentData step](#) section in [Chapter 69: Persistence](#).

▼ Starting a project from outside Matrox Design Assistant

After deploying projects to a platform, you can run and manage those projects from the **Matrox Design Assistant configuration** portal of the runtime platform. Anyone with access to the platform's portal pages can run a project.

- Open the portal by typing the name or IP address of the runtime platform into the address bar of your web browser. Format the name as an address, that is, [http://name](#).
- You will be brought to the [HOME](#) portal page. Every project deployed on your platform is listed. If you have set a project to automatically start upon platform initialization, you will see it noted as a "Startup Project".
- Click on the **Start** button next to the project you want to run.
- Optionally, click on the presented **Open Operator View** button. The operator view will be displayed in your web browser.

For other information about portal pages, see [Using the Matrox Design Assistant configuration portal web pages](#).

In addition to setting the startup project options when deploying the project, you can also change them at anytime from the **STARTUP** tab of the **PROJECTS** portal page.

▼ Starting and stopping a project using REST commands

You can also start and stop projects programatically using REST commands. REST API uses the HTTP protocol. To start a project, use:

```
PUT http://<host>/Home/StartProject/<ProjectName>
```

To stop a project, use:

```
PUT http://<host>/Home/StopProject/<ProjectName>
```

You can also use REST commands to inquire the running state of a runtime project. To do this, use the command:

```
GET http://<host>/Home/GetRunningStatus/<ProjectName>
```

The results format will depend on the format specified, the default being no format. If JSON format is specified, the result will be true or false. Other formats will return 0 or 1.

You can easily try out HTTP URLs using the command line tool Curl. The following is an example of how to start a project using Curl commands:

```
curl -X PUT -H "Content-Length: 0" "http://<host>/Home/StartProject/<ProjectName>"
```

The following is an example of using a Curl command to inquire the state of a running project, specifying the JSON results format:

```
curl -X GET -H "Content-Length: 0" "http://localhost/Home/GetRunningStatus/<ProjectName>?format=json"
-> false
```

You can also queue a validation to a validation server using REST commands. For more information on how to validate a project with REST commands, see the [Procedure for using the Project Change Validator](#) section in [Chapter 74: Project Change Validator](#).

▼ Kiosk mode

In kiosk mode, the operator view window is opened full screen while other GUI elements of the operating system and web browser are hidden from view (for example, the controls for closing and minimizing the window). Kiosk mode is typically used to prevent users from inadvertently closing the operator view or changing operating system settings.

Note that some operating system keyboard commands are still accessible in kiosk mode. Some keyboard commands can allow the user to exit kiosk mode. Kiosk mode should therefore not be considered a security feature.

To limit the access that the operator has to a platform, a typical configuration is to set a startup project and display its operator view in Kiosk mode on the browser of the target platform. The operator will only be able to see the operator view of the project. A platform can have multiple startup projects, but only one can be running in kiosk mode. As such, this setup works well when a platform will only run one project.

You must open the portal page for your PC platform or smart camera using a remote computer if you want to be able to access the operator views of other projects, or otherwise control the platform beyond starting or stopping projects remotely.

To leave kiosk mode, you must open a web browser on a remote computer and access the **STARTUP** tab on the **PROJECTS** portal page. Deselect the **Kiosk Mode** option for the project. You must reboot the runtime platform for this change to take effect. Alternatively, you can stop your project and then re-deploy it with kiosk mode deselected.

▼ Preventing runtime errors

When running a project, errors can occur. When creating your project, you should try to avoid errors and deal with errors when they happen.

You can analyze your project's execution for debugging purposes with the **MONITORING** portal page in the **Matrox Design Assistant configuration** portal. The Monitoring portal page allows you to view the current state of a running project, past inspections, and various statistics. For more information on how to use the Monitoring portal page to deal with errors, see [Using the monitoring portal page](#).

The **Status** step and **Error** step are used to handle runtime errors without causing a project to terminate. By default, all new projects start with a **Status** step, which typically should not be removed from the end of the flowchart. For more information on the **Status** step, see the [Procedure for using the Status step and Error step](#) section in [Chapter 27: Flow control steps](#).

To test communication behavior without connecting to a real PLC, you can enable PLC Emulation mode in the tab for the communication protocol you are using in the **COMMS** portal page of the **Matrox Design Assistant configuration** portal. For more information, see [PLC emulation](#).


Using the Matrox Design Assistant configuration portal

The **Matrox Design Assistant configuration** portal is a set of web pages, referred to as portal pages, resident on the runtime platform.




You can use the portal pages of the **Matrox Design Assistant configuration** portal (for both a PC and a supported Matrox Iris smart camera platform) to:


- Manage projects installed on your runtime platform (specifically, to run projects, show the operator view, and delete unused projects to free memory space on your runtime platform). For more information, see the [Managing projects on your runtime platform](#) subsection of this section.
- Monitor the running projects and display which steps and flowcharts are running in a timeline that also shows PLC communication events and Operator View events and timing statistics with the [Monitoring](#) tab.
- View images from your camera.
- View EtherNet/IP assemblies, PROFINET modules, CC-Link module labels, and optionally emulate **Quick Comm DataFromPLC** (see the [Portal pages for monitoring communication and emulating a PLC](#) section in [Chapter 51: Communication overview](#) for more information).
- View Modbus data tables and auxiliary I/O signals.

- Emulate transmission of data and triggers from a PLC to your runtime platform using a supported communication protocol.
- Define calibrations and, for color cameras, define white balance settings.
- View and manage options accessible using the **Administration**  button. These include **Specify Robot interface IP address**, **View OPC-UA endpoints**, and **Modify the default Documents folder for saving images and text files**.
- Manage some security features.

A supported Matrox Iris smart camera's configuration portal (such as the Matrox Iris GTR/GTX) has additional platform-specific administrative and configuration pages (for example, the **DEVICE** portal page), accessible using the **Administration**  button. For more information about additional portal pages on the Matrox Iris GTR/GTX smart camera, see the [Portal pages of your Matrox Iris GTR/GTX](#) section in [Appendix E: Matrox Iris GTR/GTX](#).

▼ Accessing the Matrox Design Assistant configuration portal

To access the **Matrox Design Assistant configuration** portal of the currently connected runtime platform from Matrox Design Assistant, click on the **Show portal in browser**  toolbar button of the **Platform** toolbar.

To view the local **Matrox Design Assistant configuration** portal of a PC runtime platform, right-click on the DA Agent notification icon () in the runtime platform's taskbar, and then select **Go to the Design Assistant portal webpage**. Alternatively, you can access the portal of a runtime platform by typing **localhost** in the address bar of a web browser on the runtime platform.

To access the **Matrox Design Assistant configuration** portal of any runtime platform on your local network from a web browser, go to the address **http://Auto_Net_Name**, where *Auto_Net_Name* is the name or IP address of the runtime platform that you want to access. The default name of a Matrox smart camera is written on a sticker on its side.

▼ Managing projects on your runtime platform

If a project is running (or has recently run) on your runtime platform, a box will appear on the portal page with the relevant information about the project. The **HOME** portal page allows you to terminate a project and run projects previously deployed to your platform.

▼ Deleting projects

You can manage the storage space on your runtime platform by deleting projects (stored on your runtime platform) that are not in regular use. You can delete these projects using the **PROJECTS** portal page by clicking on the **Delete project** button that appears after expanding the project. Additional information about the project is also provided on this portal page. You should ensure that you have a copy of the project on your development computer before deleting a project from the runtime platform. You can download a project from a connected runtime platform to your development computer, using the **Import Project** menu item in the **Quick Start** tab.

Note that you cannot delete a project if it is currently running.

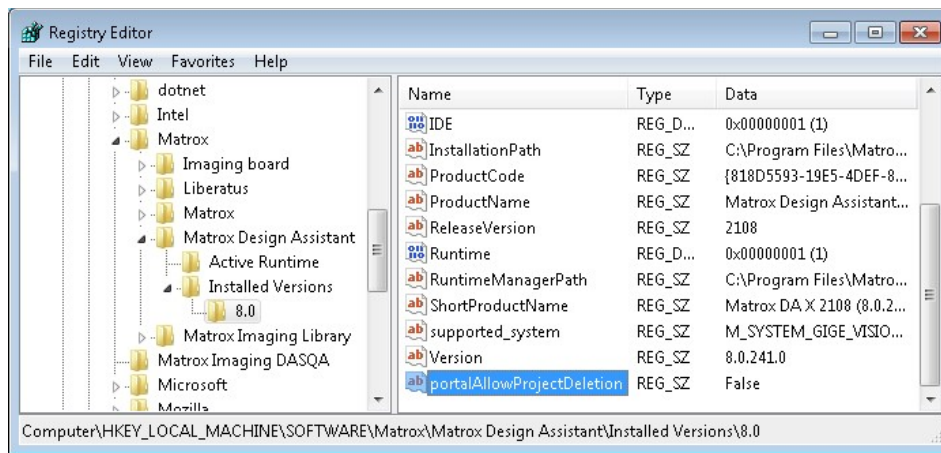
▼ Disable project deletion from portal page

On Microsoft Windows-based platforms, you can disable the deletion of deployed projects from the portal page by adding a Windows registry key, which is useful when you want to prevent operators from unintentionally deleting a project. If you are using a Matrox Iris GTR/GTX, see the [Disabling project deletion from the portal page](#) subsection of the [Portal pages of your Matrox Iris GTR/GTX](#) section in [Appendix E: Matrox Iris GTR/GTX](#).

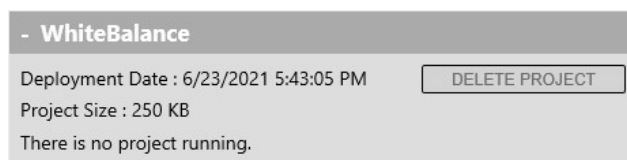
Changes made to the registry can potentially render the operating system unstable or inoperable; you should be careful not to make any changes other than those outlined below.

To add the registry key, do the following:

1. Press **Win+R** and enter **regedit** in the **Run** dialog that pops up. This will open up the Registry Editor.
2. In the Registry Editor, navigate to *Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Matrox\Matrox Design Assistant\Installed Versions\<version_number>*, where *<version_number>* is the number of your Matrox Design Assistant version (for example, 8.0).
3. In the toolbar of the Registry Editor, select **Edit**, **New**, and finally, **String value**. Name the key **portalAllowProjectDeletion**.
4. Right-click on **portalAllowProjectDeletion** and select **Modify**.
5. In the dialog that appears, set the value to **False**.
6. Close the Registry Editor and restart the Matrox Design Assistant Runtime Environment to apply the change.



To verify that the change was effected, go to the **PROJECTS** portal page and ensure that the **Delete project** button is grayed out, as shown below. If it is not, restart your computer.



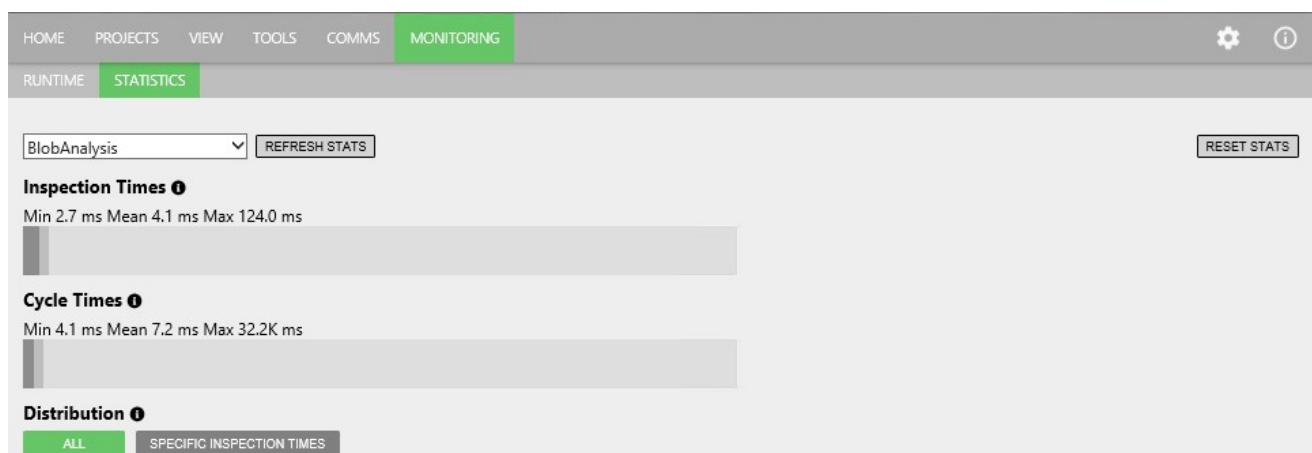
To manage storage space when project deletion from the portal page is disabled, navigate to the path `C:\ProgramData\Matrox Design Assistant\<version_number>`, where `<version_number>` is the number of your Matrox Design Assistant version (for example, 8.0), and manually delete the project from both the **Projects** and **Web** folders. Be warned that when using this method to delete projects, there are no safeguards in place that prevent you from deleting a project that is currently running.

▼ Using the view utility

To view images from your camera so that you can adjust focus and exposure, as well as specify file naming options, use the view utility, accessible from the **VIEW** portal page. Click the **Start** button to open the utility and view images from your camera. Set the exposure time and focus as required. You can also specify to capture the date and time, whether to number your saved files, and/or directly save images.

▼ Using the monitoring utility

The **MONITORING** portal page collects and displays details of recent project activity for projects in the runtime. It also collects long-term statistics.



▼ Runtime Tab

To view the current state of the inspection of a running project, click on the **Runtime** tab. Use the dropdown menu to choose which project to view, and click **Refresh** to show its information. The **Runtime** tab shows the time and name of each flowchart step that has run and its time of execution. The blue bars to the right of the chart display the time it took to complete each event or step in milliseconds. If the project is receiving triggers or being configured in the operator view pages, click the **Refresh** button or click the F5 key to display the most recent status.

Loop shows the current loop count. By default, the last 2 loops are displayed, and 10 loops are stored. You can modify how many loops are displayed or stored by clicking the

Configure gear button next to Loop. To view the activity of previous loops, click the up arrow next to the loop identifier.

The **Time** column displays the InspectionStart point and InspectionEnd point which is represented by a vertical line that extends to the location of InspectionEnd. Each inspection runs inside a loop.

The **Flowchart** column shows where the project is in the flowchart. Branching steps, such as **Condition** or **Switch**, will indicate which branch was taken. When flowcharts are called via a flowchart step they appear inline, indented for each nested call. The step that is currently running will have a yellow highlight; this is particularly useful for **Camera** step and triggers, because you can see if the project is currently waiting for a trigger.

The **Events** column displays events associated with operator view input elements or with PLC events. Image queue activity is also shown in the Events column. A + symbol indicates when a triggered image is added to the queue for the named physical camera by the acquisition thread. A – symbol indicates when a Camera step removes the image from the queue. The number indicates the number of images acquired by that camera.

For operator view events, you can hover over the name of a button to display the name of the operator view page on which it is located. Adjustments made to an editable region are also displayed as an event. You can hover over a **Display** event to show the before and after values of the region. If an operator view input event calls an event flowchart, you can display the corresponding execution of the event flowchart by clicking on the name of the event. To display the first execution of the event flowchart, click on the event flowchart's name at the top. The up and down arrows select a previous or subsequent execution for display. To return to the main flowchart, click on its name at the top.

When **Quick Comm** is enabled, the events column shows information received from the PLC and data sent to the PLC. If the flowchart is waiting for a response, the last item will blink yellow. If you are testing and do not have a PLC available, the **COMMS** portal page supports a PLC Emulation mode where you can change the values in **DataFromPLC** fields that would normally be supplied by the PLC. Quick Comm handshake signals will be automatically provided when emulation is enabled.

▼ Statistics Tab

To view statistics about the runtime, click on the **Statistics** tab. This page displays the distribution of inspection times over all the main loops. The **Inspection Times** bar show the mean, median, and average of the time between the Inspection Start to Inspection End points. The inspection time is the sum of the duration of all Steps that were executed and the duration of all publishing groups. This is typically the active processing from the moment the new image data is received to the bottom of the main loop. It does not include the time waiting for a trigger to arrive, nor the time spent in event handling flowcharts.

Cycle Times display the time it took from the Inspection End to the next Inspection End. This includes all waiting and event handling, which can be useful in cases where you need to see how long you have been waiting for a trigger. The values are likely to be more widely varying, especially if there is human interaction in training and setup pages while the project is offline from the PLC.

Distribution displays a bar graph for each inspection time logged as a histogram, with each inspection sorted into a bin based on the time it took to complete an inspection. Below the distribution are the steps in descending order. You can click on a bar to enter **Specific Times Mode** and view statistics for a single bin of the distribution graph, or click **All** to see statistics across all inspections. Hovering over a bar displays the last 10 inspections for the inspection time: Index is the inspection number, and Time is the time at which the inspection ended. The first inspection for the bin of inspection times will also be shown. Typically the longest inspection is the first one, since some initialization and allocation happen the first time some code is called. This graph can be used to find information such as the shortest and longest loop by hovering over the bar in the smallest or largest bin, respectively.

▼ Network folders and persistent connections with a supported Matrox smart camera

A project running on a supported Matrox smart camera might require access to a folder on the network (for example, if using the **ImageWriter** step, the **TextReader** step, or the **TextWriter** step). For more information about accessing a network folder, and how to validate your domain access credentials, see [Acquiring permission to a network folder](#).

Alternatively, you can establish a persistent connection, which provides a convenient way to access a specific network folder. For information on how to specify the permission to access these folders or set up a persistent connection, see the [Reading and writing files on your Matrox Iris GTR/GTX](#) section in [Appendix E: Matrox Iris GTR/GTX](#).