# PRECISE AUTOMATION

The PreciseVision Machine Vision System

Introduction and Reference Manual

Version 2.1.1, Apr. 22, 2009
P/N: PVS0-DI-00010

Document Content

The information contained herein is the property of Precise Automation Inc., and may not be copied, photocopied, reproduced, translated, or converted to any electronic or machine-readable form in whole or in part without the prior written approval of Precise Automation Inc. The information herein is subject to change without notice and should not be construed as a commitment by Precise Automation Inc. This information is periodically reviewed and revised. Precise Automation Inc. assumes no responsibility for any errors or omissions in this document.

Trademarks

Guidance 3400, Guidance 3300, Guidance 3200, Guidance 2400, Guidance 2300, Guidance 2200, Guidance 1400, Guidance 1300, Guidance 1200, Guidance Controller, Guidance Development Environment, GDE, Guidance Development Suite, GDS, Guidance Dispense, Guidance Programming Language, GPL, Guidance System, PrecisePlace 1300, PrecisePlace 1400, PrecisePlace 2300, PrecisePlace 2400, PreciseFlex 1300, PreciseFlex 1400, PrecisePower 300, PrecisePower 500, PrecisePower 2000, PreciseVision, RIO are either registered or trademarks of Precise Automation Inc., and may be registered in the United States or in other jurisdictions including internationally. Other product names, logos, designs, titles, words or phrases mentioned within this publication may be trademarks, service marks, or trade names of Precise Automation Inc. or other entities and may be registered in certain jurisdictions including internationally.

Any trademarks from other companies used in this publication are the property of those respective companies. In particular, Visual Basic, Visual Basic 6 and Visual Basic.NET are trademarks of Microsoft Inc.

Disclaimer

PRECISE AUTOMATION INC., MAKES NO WARRANTIES, EITHER EXPRESSLY OR IMPLIED, REGARDING THE DESCRIBED PRODUCTS, THEIR MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE. THIS EXCLUSION OF IMPLIED WARRANTIES MAY NOT APPLY TO YOU. PLEASE SEE YOUR SALES AGREEMENT FOR YOUR SPECIFIC WARRANTY TERMS.

Precise Automation Inc.
727 Filip Road
Los Altos, California 94024
U.S.A.
www.preciseautomation.com

Warning Labels

The following warning and caution labels are utilized throughout this manual to convey critical information required for the safe and proper operation of the hardware and software. It is extremely important that all such labels are carefully read and complied with in full to prevent personal injury and damage to the equipment.

There are four levels of special alert notation used in this manual. In descending order of importance, they are:

DANGER: This indicates an imminently hazardous situation, which, if not avoided, will result in death or serious injury.

WARNING: This indicates a potentially hazardous situation, which, if not avoided, could result in serious injury or major damage to the equipment.

CAUTION: This indicates a situation, which, if not avoided, could result in minor injury or damage to the equipment.

NOTE: This provides supplementary information, emphasizes a point or procedure, or gives a tip for easier operation

Table Of Contents

PreciseVision Machine Vision System

PreciseVision Introduction

PreciseVision is a powerful, full featured 2D machine vision software package. This software application includes a complete toolkit with everything that is needed to acquire vision images, identify and locate randomly positioned and oriented parts, measure key features and inspect parts for specific defects. Each of the basic vision functions is provided as a type of "Tool" in the vision toolbox and is internally designed as a modern software "Object".

This software package has been designed to operate with the Guidance Programming Language, which executes on a Precise Guidance Controller, to produce flexible vision-guided motion applications. PreciseVision executes on a Windows PC system and includes a built-in interface to communicate with a Precise Controller via Ethernet. The Guidance Programming Language, GPL, includes a complementary built-in communications interface to PreciseVision and has classes and objects for easily controlling the operation of PreciseVision and retrieving vision results. A complete vision guided robot application consists of a list of vision tools executing in PreciseVision and a GPL program executing on a Guidance Controller that takes the vision results and utilizes this information to alter the actions of a robot.

PreciseVision includes a very easy-to-use visual programming interface that allows both simple and complex vision programs to be constructed by dragging-and-dropping tools into a list. Training of the tools is performed by visually positioning and sizing tools on top of a captured camera image. The parameters for each tool can also be access via a familiar "Properties Table". Since each tool is a software object, the output of one tool can be easily linked to the input of other tools. This is a very powerful capability that allows the placement and the actions of tools to be a function of the results of previously executed tools and permits the results of several tools to be combined.

To perform the actual image capturing, PreciseVision supports a number of third party Ethernet cameras and a third party Ethernet converter box that can interface to conventional RS-170 analog cameras. Vendor and part number information for purchasing this hardware and installation instructions are provided in this document for supported 3rd party hardware.

In addition to the normal mode of executing tools on a real camera image, this package also permits a sequence of images to be easily saved to disk and then subsequently played back. This allows vision tools to be tested off-line and provides a convenient means for customers to obtain assistance from Precise and its partner companies remotely.

In this document, instructions for installing PreciseVision and the supported 3rd party hardware is presented. This is followed by an overview and quick start section that introduces machine vision newcomers to PreciseVision's principal concepts and permits experienced individuals to quickly use the system. Finally, the last sections of this manual provide detailed information on the system's user interface, toolkit and camera calibration procedures.

For further information on programming a vision guided robot application in GPL, please refer to *The Guidance Programming Language, Introduction to GPL*.

Supported Hardware and Software Installation

Supported Vision Hardware

PreciseVision is a software package and does not include hardware. This software is designed to capture vision images that are transmitted over Ethernet. PreciseVision supports two 3rd party Ethernet hardware configurations: (Option 1) images generated by DALSA Ethernet Cameras or (Option 2) images generated by analog cameras that are digitized by a Pleora iPORT Ethernet converter. Option 1 is simpler and more direct, while Option 2 allows for the use of a wider variety of available cameras.

In this section, part numbers are provided to allow hardware to be directly purchased from the hardware vendors together with simplified wiring drawings to assist in connecting the components.

Option 1: DALSA Ethernet Cameras

- PreciseVision supports up to 6 DALSA Genie™ GigE Ethernet cameras. The Genie M640-1/3 GigE camera is the recommended camera model.
- DALSA cameras can be directly connected to the PC running PreciseVision or through an Ethernet switch.
- The PC must include a standard Ethernet card, but a standard GigE card is recommended for the best performance.
- For more information, visit the DALSA website at www.imaging.com. For ordering, contact Manish Shelat at DALSA Digital Imaging, phone: (510) 249-5664, email: manish.shelat@DALSA.com.

PreciseVision using DALSA Genie Camera

| Hardware Specification	
ETHERNET CAMERA	DALSA Genie M640-1/3 GigE Vision Camera, part # CR-GEN3-M6400, 640x480 pixel monochrome sensor, 60 frames/second raw frame rate. Progressive scan, 1/3" sensor. Effective frame-to-frame time depends on the exposure time which ranges from 100 microseconds to 4 seconds, controlled via software. For total time, add vision processing which is typically less than 100ms. PreciseVision supports up to 6 DALSA cameras.
MOUNTING ADAPTER	Optional DALSA mounting bracket assembly, 1/4" screw mount, part # CA-GENA-BRA00. This bracket is used for mounting the camera on a standard tripod.
CAMERA POWER	Each camera requires 4 watts, 11 VDC to 13 VDC. Option 1: If you wish to break out the camera's I/O pins, order DALSA part # CA-GENA-PS120 and the DALSA Genie I/O and power breakout cable assembly, part # CA-GENC-IOP00. Option 2: If you do not need I/O signal break out, an alternative is the Intercon PSISP10, which includes a connector compatible with the camera but no I/O signal break out, www.nortechsys.com/intercon/index.html
ETHERNET SWITCH	Standard GigE Ethernet Switch. Netgear GS605 10/100/1000, or equivalent.
ETHERNET CARD	Standard GigE PC Ethernet card. Netgear GA311 PCI Gigabit Adaptor, or equivalent.
ETHERNET CABLE	Standard Cat-5 Ethernet cable. Total of 3 required.

Option 2: Analog Cameras with Pleora iPORT

- PreciseVision supports up to 6 RS-170 or NTSC analog cameras using the Pleora Technologies iPORT Analog Video to Ethernet Converter.
- The iPORT converter is connected directly to the PC running PreciseVision, or through a 100M or GigE Ethernet switch.
- You can select from a wide variety of analog cameras.
- The PC must include a standard 100M or GigE Ethernet card.
- For more information, including a list of supported analog cameras, please contact: Pleora Technologies, phone: (613) 270-0625 email: info@pleora.com or visit the website www.pleora.com.

PreciseVision using Analog Camera with Pleora iPORT

Hardware Specification	
ANALOG VIDEO TO ETHERNET CONVERTER	iPORT Analog Video IP Engine, part #PT1000-ANL-2-6-E, supports 1 to 6 RS-170 or NTSC analog cameras. Effective frame-to-frame time is 50ms average for repeated pictures on a single camera. Time for switching between cameras of the same type is 134ms. Time for switching between cameras of different types is up to 250ms. For total time, add vision processing which is typically less than 100ms.
CONVERTER POWER	iPORT power supply with cable and connector (12 volts), part #PT1000PWR-H.
CAMERA, RECOMMENDED	Pulnix TM-250 or equivalent. For info visit www.pulnix.com. Note: Other cameras are also compatible.
CAMERA POWER	For Pulnix TM-250 camera: Pulnix PD-12UU power supply + Pulnix 12P-02S power cable.
VIDEO CABLE	Standard RG-59 coaxial video cable.
ETHERNET SWITCH	Standard 100/GigE Ethernet Switch. Netgear GS605 10/100/1000, or equivalent.
ETHERNET CARD	Standard 100/GigE PC Ethernet card. Netgear GA311 PCI Gigabit Adaptor, or equivalent.
ETHERNET CABLE	Standard Cat-5 Ethernet cable. Total of 3 required.

Installing PreciseVision on a PC

PreciseVision is distributed as a standalone application that executes on a Windows PC system. For the best results, it is suggested that this software be executed on the following equipment:

- 2.3Ghz or faster Pentium 4 PC running Windows XP
- Gigabit Ethernet (recommended and required for certain cameras), otherwise 100Mb Ethernet interface
- 512 MB of RAM
- At least 1 GB of space on the PC's disk

- A 2x CD-ROM drive interfaced to the PC

While PreciseVision can be executed by itself to develop Vision Processes, the PC must be able to communicate with a Precise Guidance Controller via Ethernet in order to run a complete vision-guided motion application. **Please see the *"Guidance System Setup and Operation, Quick Start Guide"* for instructions for configuring the PC and the Guidance Controller to communicate with each other.**

If you previously installed PreciseVision on your computer, you should un-install the old version by performing the steps below. If you are installing PreciseVision for the first time, you can skip the first set of instructions.

> » Shut down all programs that are running including virus protection programs.
>
> » Bring up the Window's Control Panel by clicking "***Start > Settings > Control Panel***".
>
> » Double click on the "***Add or Remove Programs***" selection.
>
> » In the "Add or Remove Programs" popup window, scroll down and click on "***PreciseVision***".
>
> » Click the "***Remove***" button and click on "***Yes***" to confirm the action.
>
> » Close all of the windows that you opened.

To install PreciseVision on your computer, perform the following steps:

> » Shut down all programs that are running including virus protection programs.
>
> » Insert the PreciseVision CD-ROM into your computer's CD-ROM drive. A panel should pop up that welcomes you to the Setup Wizard. If the installer does not automatically start, click "***Start > Run***", type in "***D:\setup.msi***" (where D is the CD-ROM drive), and click "***Ok***".
>
> » Follow the instructions in the Setup Wizard to install PreciseVision.   Note, PreciseVision relies upon the Microsoft .NET Framework in order to operate. This is a standard module that Microsoft provides free of charge. If the Setup Wizard detects that this module is not available, you will be asked if you wish to install the Framework now. You should respond "***Yes***". This will launch a browser to take you to the download site for the required software.

At the conclusion of this process, PreciseVision will be installed on your PC. This software by itself is sufficient to execute PreciseVision with camera images that have already been captured and saved to a disk file. But, to operate this software with a camera, **one of the camera interface installation procedures in the following sections must be performed**. In addition, in order to continue to use PreciseVision beyond the trial period, **you must register this product with Precise Automation.** The registration process is described in detailed in a following section.

Installing the DALSA Genie™ Camera Interface

If your system is equipped with a DALSA Genie™ GigE camera, the procedure described in this section must be executed to interface this device to PreciseVision. If your system does not use this camera, please skip to the next section of the documentation.

PreciseVision

Prior to executing this procedure, ensure that the following requirements are satisfied:

- The DALSA Genie™ camera must be interfaced to the PC via a 100MB or 1000MB (Gigabit) Ethernet card. A 1000MB card will produce the best performance although a 100MB card will produce acceptable performance for many applications.

The first step in the process is to install the DALSA device driver and its configuration software.

> » Insert the PreciseVision CD-ROM into your computer's CD-ROM drive.
>
> » In the *"DALSA/SaperaLT/"* folder, double click *"Setup.exe"* to execute the installation procedure.
>
> » Follow the instructions in the Setup Wizard to install the driver.
>
> » In the *"DALSA/Genie/"* folder, double click *"Genie_1.nn.nn.nnnn Release.exe"* to execute the installation procedure.
>
> » Follow the instructions in the Setup Wizard to install the driver.
>
> » After installing the Genie drivers, the PC must be rebooted.

After rebooting the PC, you should see a small camera icon. in the task bar in the right hand corner but with a red 'X' over it.

Plug the Genie camera into the network. After a period of time the 'X' will disappear. If you hover over the icon, the tool tip will indicate the number of Genie cameras that have been located. If no cameras are located, please see the following "DALSA Genie Installation Trouble Shooting" table.

DALSA Genie Installation Trouble Shooting	
Trouble	**Possible Solutions**
When the PC boots up it is not displaying the little camera icon in the system tray.	The Genie driver did not properly install. If you have VPN software installed, try removing this and re-installing the Genie driver.
The camera icon is displayed but always has a red 'X' over it.	If you have multiple network cards, try opening up the 'DALSA Network Configuration Tool'. This is found in Start -> DALSA -> Sapera Network Imaging Package.

Select the 'Network' Adapter you are using with the camera. Select the 'DHCP/LLA' mode in the Device Configuration section. |

If you are still having difficulty with the operation of your DALSA camera, please see the FAQ chapter of this manual for additional step-by-step trouble shooting instructions.

Once the camera is properly interfaced to the PC, one final step remains. It is possible that the firmware that shipped with your DALSA Genie camera may not match the PC driver that you just installed. If the firmware and the driver do not match, the camera may not operate reliably. To determine if the camera firmware needs to be updated, perform the following steps:

6

> » Ensure that the camera is properly communicating with the PC per the instructions above.
>
> » Open the DALSA Device Manager firmware update dialog. This is accomplished from the Window's Start menu by clicking: **"Start > Programs > DALSA > Genie > Firmware Update**".

A list of available cameras will be displayed as shown in the following picture.
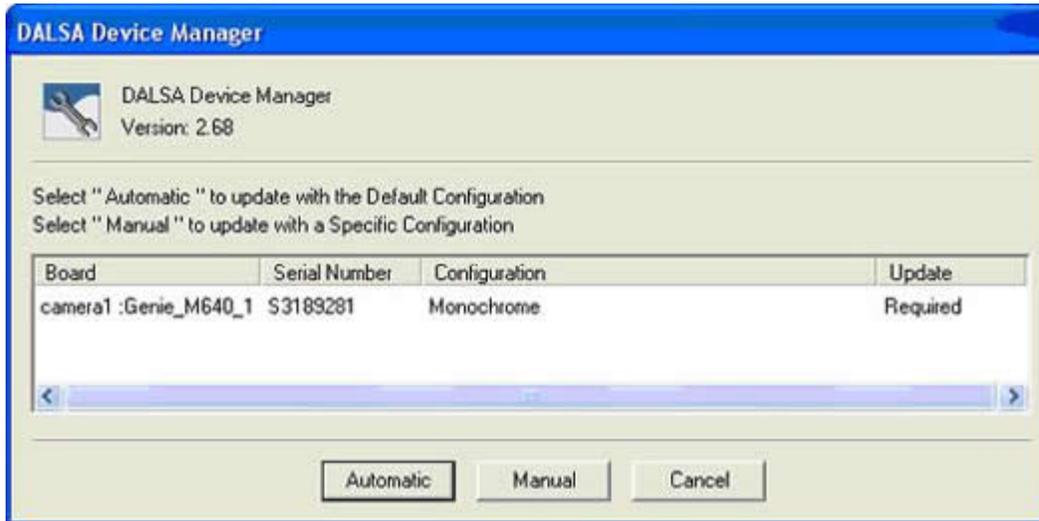
The right most column labeled "Update" indicates the firmware status of each camera.

> » If the "Update" status is "Not Required", the camera firmware matches the driver and the following steps can be skipped.
>
> » If "Required" is indicated, the camera firmware must be updated for proper operation. To update the firmware, click on the "**Automatic**" button.
>
> » Allow up to 3 min. for the camera to update. **Do NOT unplug the camera during this time.**
>
> » After the camera re-initializes the LED on the camera next to the power connector should return to a solid BLUE state. This indicates that the process is completed and you can now exit from the DALSA Device Manager.

This completes the DALSA Genie™ camera setup. PreciseVision can now acquire camera images via this camera(s).

Installing the Pleora iPORT Camera Interface

If your system is equipped with a Pleora iPORT Analog Video IP Engine to convert analog camera images to Ethernet messages, the procedure described in this section must be executed to interface this device to PreciseVision. If your system does not use this camera converter, please skip to the next section of the documentation.

Prior to executing this procedure, ensure that the following requirements are satisfied:

PreciseVision

- The Pleora iPORT must be interfaced to the PC via a 100MB or 1000MB (Gigabit) Ethernet card. A 1000MB card will produce the best performance although a 100MB card will produce acceptable performance for many applications.

The first step in the process is to install the Pleora device driver and its associated configuration software.

» Insert the PreciseVision CD-ROM into your computer's CD-ROM drive.

» In the *"Pleora"* folder, double click *"Setup.exe"* to execute the installation procedure.

» Follow the instructions in the Setup Wizard to install the driver. When asked what type of installation should be performed, select *"Runtime Drivers"*.

Next, you must execute the Pleora iPORT camera configuration software to configure a camera for use with PreciseVision.

» Start the iPORT camera configuration utility by clicking "*Start > Pleora Technologies Inc > iPORT Software > Launch Coyote Application*".

» When the application launches, it will open an Options dialog box. Click *"Ok"* to continue to the "Coyote" main application form that should appear as follows.
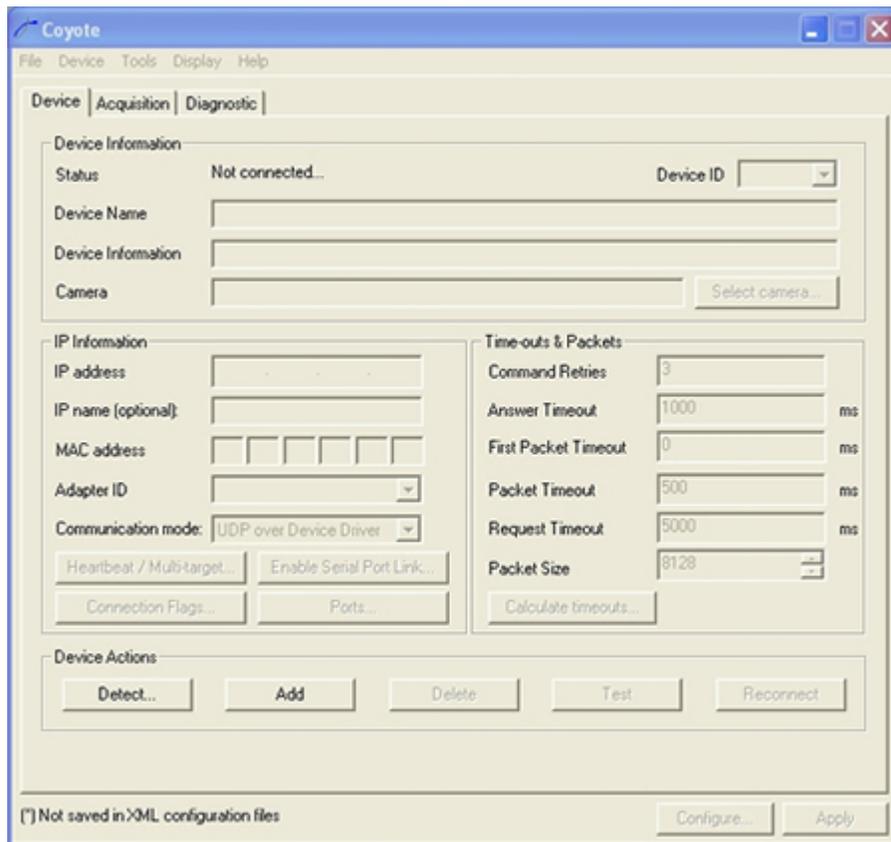
» Click on the *"Detect"* button to open the "Network Device Finder" panel. This panel

automatically searches the network for available iPORT camera interfaces.
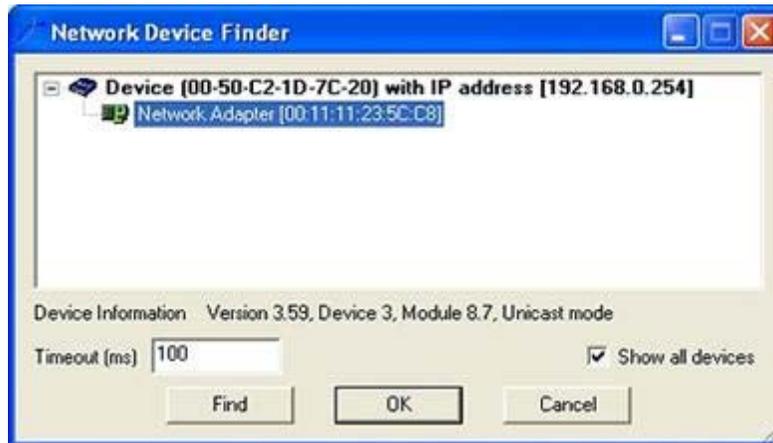
» Highlight the *"Network Adapter"* for the camera interface.

» Click on the *"OK"* button.

» If the iPORT does not have a unique IP address, an additional prompt may be displayed that assigns the iPORT an IP address. If so, click *"OK"* to accept the assigned IP address.

The Coyote main panel should now have information displayed in its "Device Information" section.
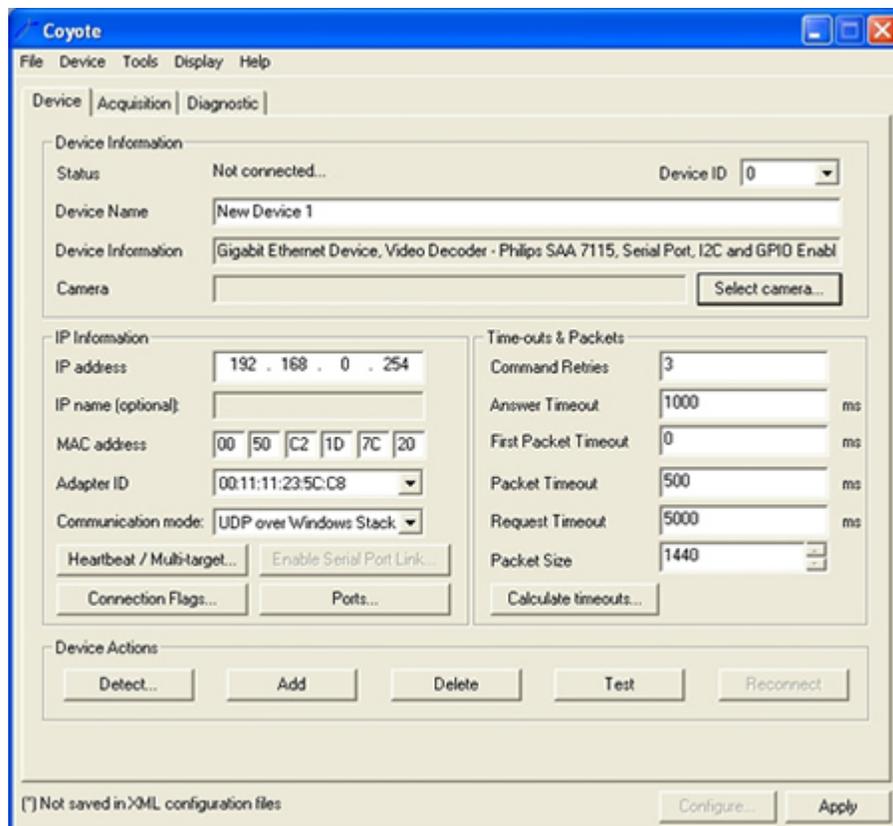
» Click the **"Select camera..."** button to open the "Select Camera" dialog as shown below.
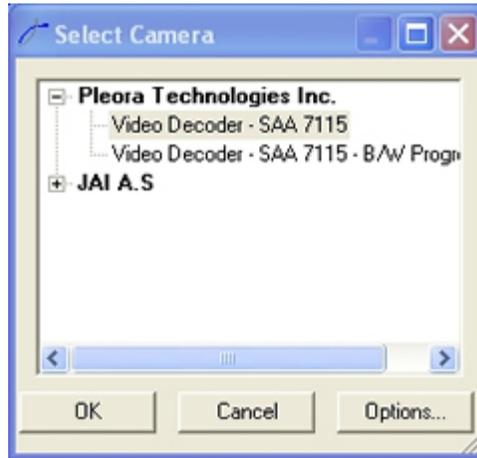
» Highlight the **"Video Decoder - SAA 7115"** under the "Pleora Technologies Inc.".

» Click the **"OK"** button to close this panel and to return to the main Coyote panel.

» In the main Coyote panel, click on the **"Acquisition"** tab to display the information shown below.
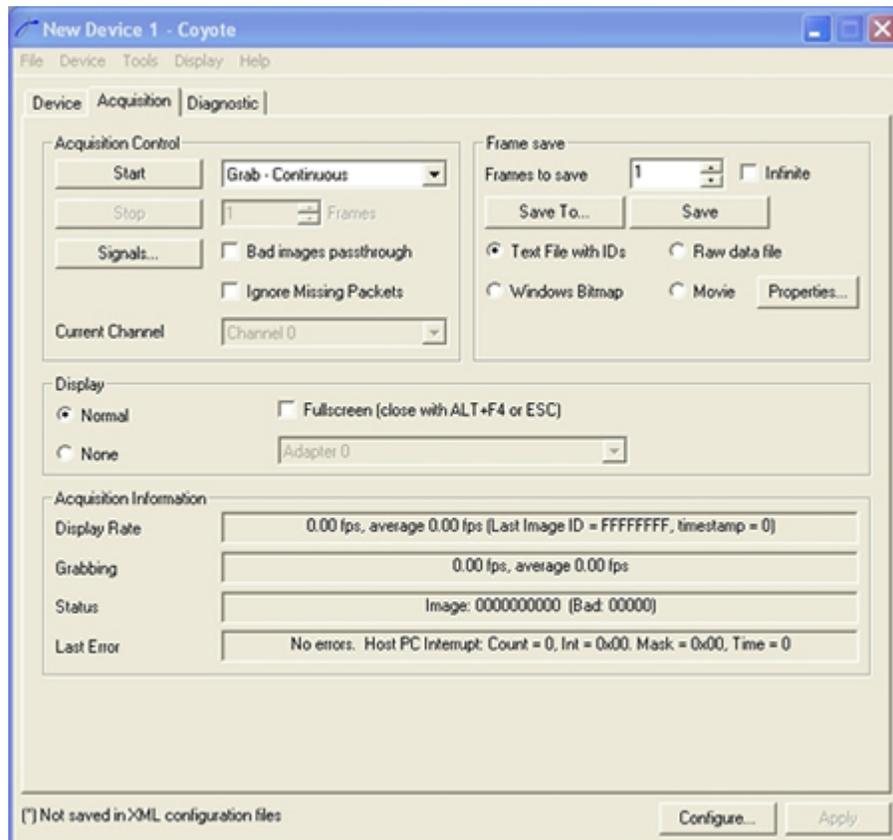
» Click on the *"Configure..."* button at the bottom of this page to open up the "Advanced Configuration" dialog panel that is pictured below.
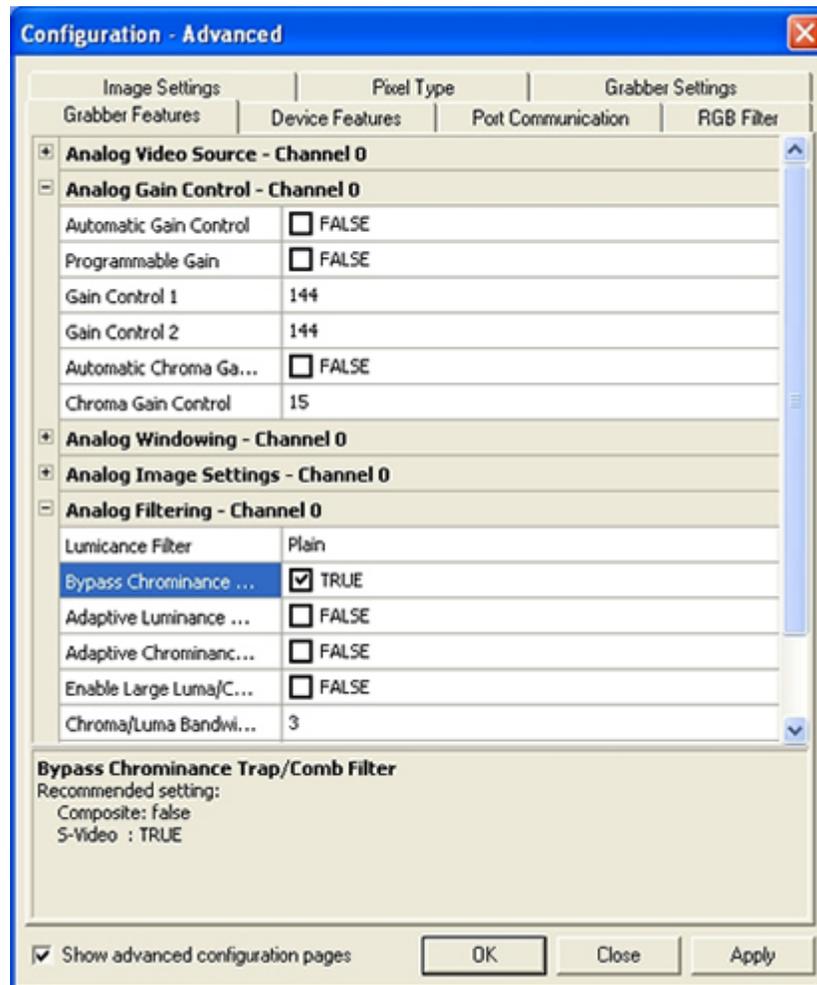
» In the "Analog Gain Control" section, *uncheck the "Automatic Gain Control" setting*. For most vision applications, it is undesirable to have the gain of the camera automatically adjusted. Unchecking this box disables the iPort from automatically handling the gain of the image.

» In the "Analog Filtering" section, *check "Bypass Chrominance Trap/Comb Filter"*. This filter causes the image to be slightly blurry and should not be executed for machine vision type applications.

» Click the *"OK"* button to return to the main form.

» To test if your camera hardware is working properly, in the main form under the *"Acquisition" tab*, click the *"Start"* button. You should see a live video image in a window.

» To return to the main form, click on the *"Stop"* button.

» To save the configuration data, in the main form select *"File > Save As"*. The data file must be placed in the root directory of your PC as *"C:\iPortConfig.xml"* in order for PreciseVision to

> locate this data.
>
> » **"Exit"** the Coyote application.

This completes the Pleora iPORT camera converter configuration setup. PreciseVision can now acquire camera images via the the iPORT.

Activating PreciseVision

To begin using PreciseVision, you must start the application on the PC.

> » To launch PreciseVision, click on **"Start > Programs > Precise Automation > PV xxx > PreciseVision".** The first time PreciseVision begins execution, you will see the following popup.
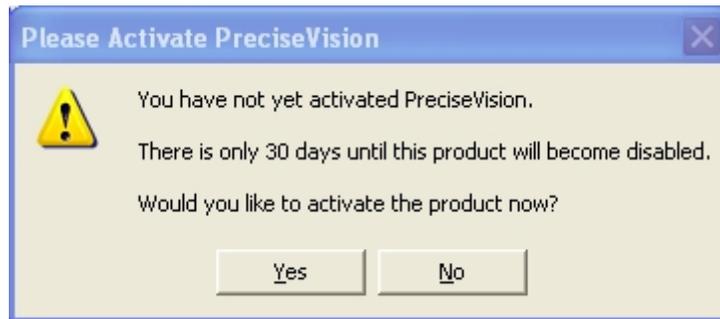
The license that you purchased for PreciseVision permits you to use this software to build and execute Vision Processes on a single PC. To complete the software activation process, you must send information concerning the PC to Precise Automation. To allow time for sending the information and receiving a reply, PreciseVision is fully functional for 30 days without being activated.

If for some reason, you do not wish to initiate the activation process now, this popup will be displayed each time you start PreciseVision. To re-display the activiation popup from within PreciseVision, click on **"Help > Product Activation"**.

> » To initiate the activation process, click **"Yes"** in the popup above. This will display the following activation popup.
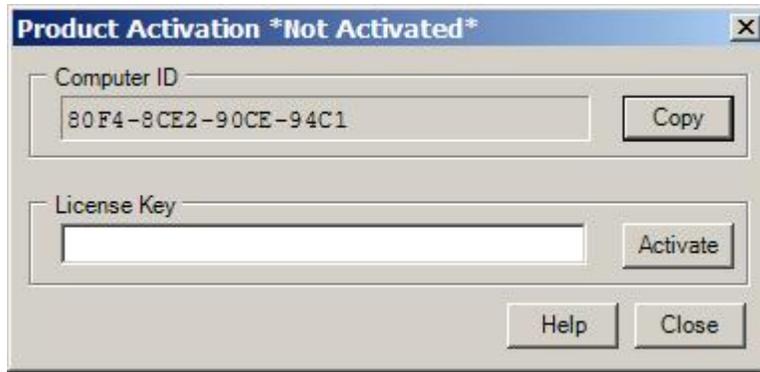
This popup displays the "Computer ID" that you need to submit to Precise as part of the activation process.

» To obtain a license key to activate PreciseVision, please send an email to Precise Automation that contains the following information:

    To: sales@preciseautomation.com
    Subject: PreciseVision License Key Request

    Customer Name: <your name here, optional>
    Customer Company: <your company name here>
    Telephone Number: <your phone number here, optional>

    PreciseVision Serial Number: PVS0-0xxx-xxxxxx
    Computer ID Number: xxxx-xxxx-xxxx-xxxx

» You can obtain the PreciseVision Serial Number from the PreciseVision License Certificate that you received with your order. Below is a picture of the top portion of a typical PreciseVision License with the serial number circled in red.
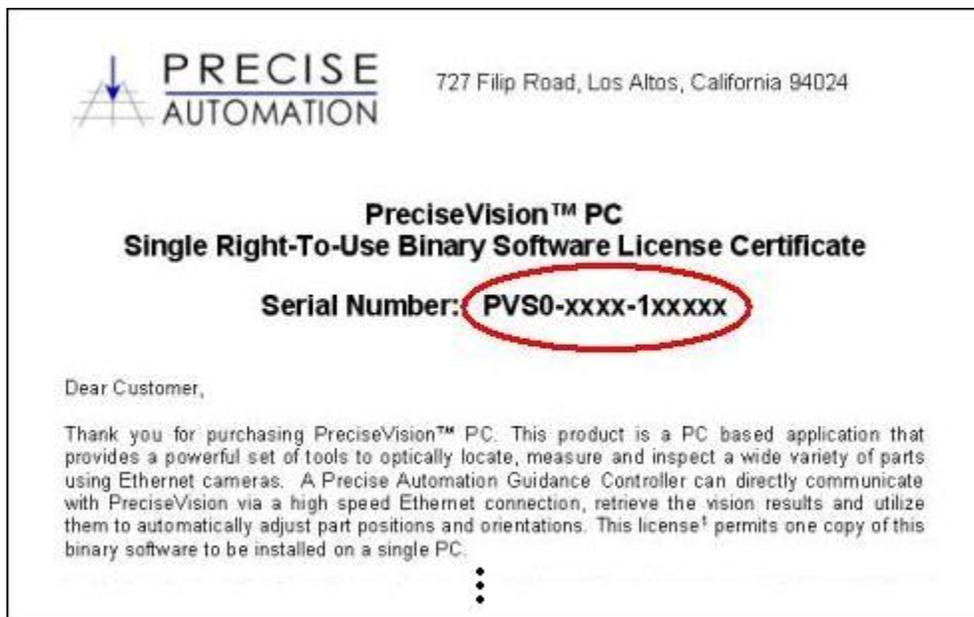
> » The Computer ID number should be copied from the Product Activation popup.
>
> » As a convenience, if you are viewing this help in the *Precise Documentation Library*, you can click on the following link to get an email template: sales@preciseautomation.com.
>
> » Alternatively, if you do not have email access, please Fax this information to 408-516-8348 together with your own fax number.

Your name and your phone number are optional. However, we recommend that you provide this information in case there is a problem with your license and we need to get in touch with you. A sample email should look as follows.

In response to your request, you will receive a license key.

> » When you receive your license key, access the Product Activation popup by restarting PreciseVision or clicking in the PreciseVision menu item *"Help > Product Activation"*.
>
> » Enter the license key in the *"License Key"* box and press *"Activate"*.

PreciseVision is now activated.

Configuring Communication Between the PC and the Controller

In order for PreciseVision to exchange information with a Guidance Controller, each system must know the Ethernet IP address of the other unit. During normal runtime execution, the Guidance Controller initiates all communications and so it must know where to send its commands. During the Robot Vision Camera Calibration procedure, PreciseVision needs to know where to send its commands to read the current robot position.

Configuring PreciseVision to Read Positions from a Controller

To permit PreciseVision to query a Guidance Controller for the current position of the robot, the IP address of the controller must be defined in the system Preferences.

> » Launch PreciseVision by clicking on *"Start > Programs > Precise Automation > PV xxx > PreciseVision"*.
>
> » Open the panel for setting the controller IP address by selecting *"File > Preferences > Client Settings"*.

The panel for setting the IP address should resemble the following:
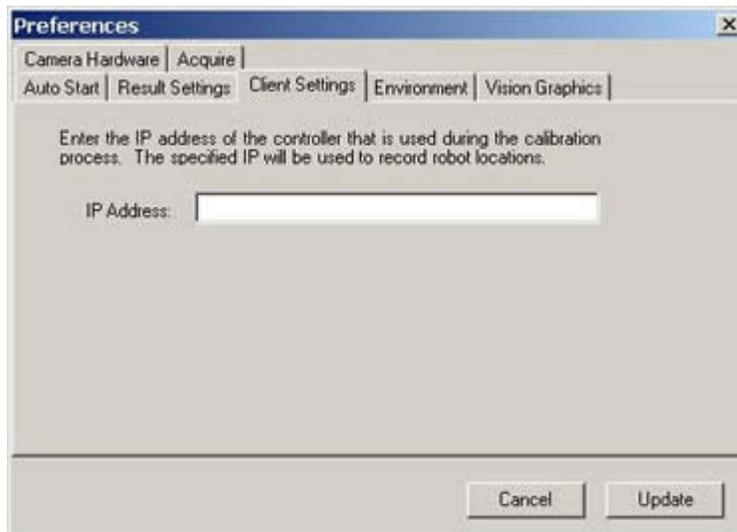
> » Enter the Guidance Controllers Ethernet IP address and press the *"Update"* button.

Configuring a Controller to Send Commands to PreciseVision

Whenever the Guidance Controller issues a command to PreciseVision, the controller automatically establishes an Ethernet connection utilizing an IP address that is specified in the controller's Parameter Database. This database is stored in the controller's flash disk and is preserved even if the controller is turned off and restarted. To modify this database value, perform the following operations.

> » Please see the *"Guidance System Setup and Operation, Quick Start Guide"* and follow the instructions for bringing up the web based Guidance Operator Interface for the controller.
>
> » In the web interface, open the database page for the vision IP address, *"Setup > Parameter Database > Controller > Network"*.

The web interface page should look like the following:
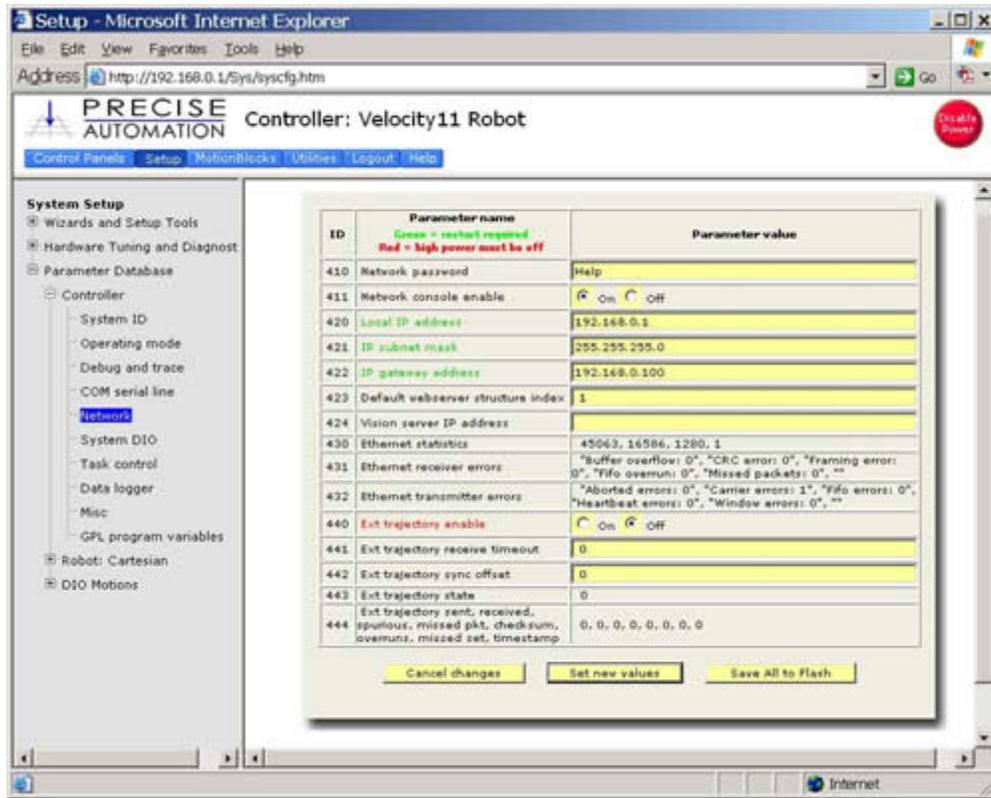
PreciseVision

> » Enter the IP address of the PC executing PreciseVision into the value for the **"Vision server IP address (DataID 424)"**.
>
> » Press the "**Set New Values**" button to store this setting into memory.
>
> » Press the "**Save All to Flash**" button to store this setting in the flash disk. This ensures that this setting will remain in effect if the controller is restarted. The button will flash for 10-30 seconds as the data is being written. **DO NOT TURN OFF YOUR CONTROLLER WHILE THE BUTTON IS BLINKING SINCE THIS MAY CORRUPT THE FLASH DISK.**

The controller is now setup to send commands to PreciseVision.   It is not necessary to restart the controller.

Overview and Quick Start

PreciseVision Overview

This Overview and Quick Start section introduces PreciseVision's principal concepts, terminology and features. For newcomers to machine vision, this will provide background information prior to reading the detailed descriptions contained in the following chapters. For individuals with experience applying machine vision, the information in this section should be sufficient for you to start using the system.

PreciseVision has been designed to operate in combination with a Precise Guidance Controller and its embedded Guidance Programming Language (GPL) to easily produce flexible and powerful vision-guided motion applications. The following diagram illustrates the key software components in a vision guidance application.
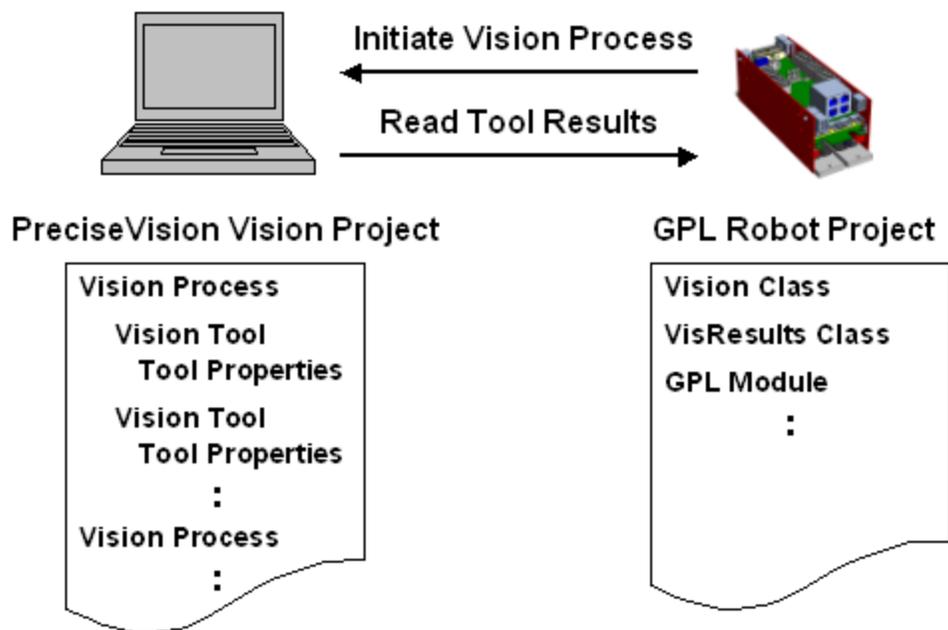
In PreciseVision, each fundamental vision runtime function is represented as a type of **"Vision Tool"** and the collection of all possible types of tools is referred to as the **"Vision Toolkit"**. For example, the function that captures a picture from a camera and stores the result in a frame buffer is the "Acquisition Tool"; the operation that fits a line to a set of edge points is the "Line Fitter Tool"; and the function that uses an advanced, patented algorithm to identify and locate parts is the "Finder Tool".

To perform an element of a vision application, any number of Vision Tools can be combined into a **"Vision Process"**. Normally, an Acquisition Tool is the first tool in each Vision Process. This is followed by one or more tools to perform part location, metrology or inspection operations. Multiple copies of the same type of tool can be included within the same Vision Process. The individual characteristics of each

tool (i.e. its size, position, algorithm parameters) are stored in Object **"Property Lists"**. After a tool is executed, its output data is stored in its Property List in a **"Tool Results"** section. To simplify their definition, tools can be positioned and sized by dragging their graphical handles on top of the display of a camera image.

A Vision Process is the basic unit of execution that can be controlled by a GPL procedure. Using a built-in **"Vision Class"**, a GPL procedure running on a Precise Guidance Controller can start the execution of a Vision Process in PreciseVision via Ethernet and monitor its progress. When the Vision Process completes execution, the GPL procedure can retrieve the Tool Results of any tool within the executed Vision Process by accessing the built-in GPL **"VisResults Class"**.

In most cases, only a single Vision Process is executed by a GPL procedure in order to perform the complete machine vision task. Typically, this Vision Process will take a picture and then utilize Vision Tools to locate a part and validate some key features or dimension. However, if a more complex machine vision operation is required, a GPL procedure can execute any number of Vision Processes. For convenience, multiple Vision Processes can be stored in a **"Vision Project"** in PreciseVision. At any given time, only one Vision Project can be loaded into PreciseVision and only one of its Vision Processes can be executing.

In order for PreciseVision to return Tool Results in meaningful units, e.g. millimeters, and in an appropriate coordinate system, any camera used for vision guidance must first be "Calibrated". As a convenience, PreciseVision includes a number of **"Camera Calibration"** methods that are described in a later section. Once a camera is calibrated, all data retrieved by a GPL procedure via the VisResults Class will be in millimeters and in the world coordinate system of the robot.

In the following sections, the Vision Tools are briefly described along with the methods for creating and editing Vision Projects, Vision Processes, and tools with their associated Property Lists.

Vision Toolkit Summary

In order to effectively analyze images, PreciseVision includes a full set of vision tools for acquiring images, locating and identifying parts, performing metrology operations and executing visual inspection of key features. In this section, the function of each tool is briefly summarized and a picture displaying how the tool appears on the Camera Display Window is presented.

An important feature of PreciseVision is that the output of one tool can be easily utilized as the input for another tool. For example, after the Finder Tool locates one or more parts, a tool that measures the distance between two edges can be easily placed relative to the located part. This measurement tool can then test if a critical dimension of the part is within the desired tolerance. Since this measurement is performed relative to the output of the Finder, it will operate correctly even if the part's position or orientation changes. Furthermore, if the Finder locates more than one part, the measurement tool will be automatically applied to each located part. This ability allows complex measurements and tests to be easily built-up from the basic tools that are provided.

For a complete description of each tool, please read the "Vision Toolkit" chapter. (Note: The algorithms for the complete set of tools has been developed and will be released in future upgrades to PreciseVision during 2007. If you cannot find a tool that you require, please contact Precise Automation.)

Graphical Display	Vision Tool	Description
N/A	**Acquisition Tool**	Captures an image from a camera or loads

		it from a disk file, and stores the image into a frame buffer.
	Arc Fitter Tool	Searches a specified region for arc edge points and returns the arc or circle that best fits the edges. Returns both the arc center point and radius.
	Clear Grip Tool	Verifies that there is no obstruction within the bounds of a defined window. Typically used to confirm that gripping a part will not result in a collision.
	Color Pixel Window Tool	Tests if the general color of a rectangular or circular region matches a trained color within a specified tolerance.
	Connectivity Tool	Searches a specified region for binary blobs and returns the centroid and other features of each blob. In addition, this tool can return data that defines the perimeter of any located blob.
	Edge Locator Tool	Detects edge points with sub-pixel accuracy along a linear path in a vision image and returns their positions. Alternately, this tool can return an array of the intensities along the linear path or a histogram of their values.
	Finder Tool	Most powerful vision tool that identifies randomly placed parts in a camera image and returns their position, orientation and scaling to within sub-pixel accuracy.
	Fixed Frame Tool	Places a reference frame at a fixed image coordinate or at a constant offset relative to another vision object. Can optionally index the frame in an X and/or Y grid pattern to repeatedly execute any linked tools.
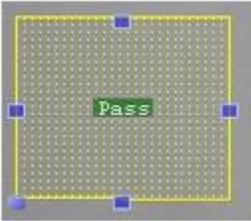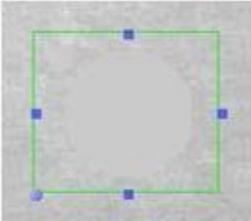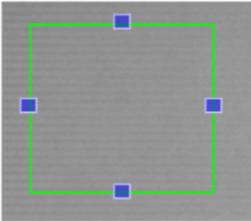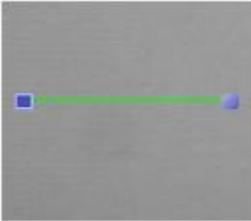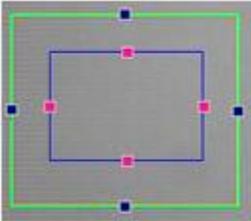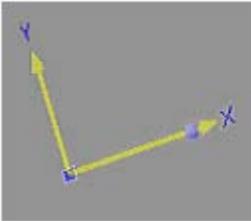
	Line Fitter Tool	Searches a specified region for edge points and returns the line that best fits the edges.
	Line-Line Frame Tool	Determines the intersection of two lines defined by vision tools and returns a reference frame.
	Pixel Window Tool	Counts edges or binary pixels or collects gray-scale statistics within a rotated rectangular or circular region. Used for quickly detecting the presence of features or collecting general intensity information about a region.
	Point-Line Frame Tool	Takes a point and a line from two vision tools and returns a reference frame.
	Point-Point Line Tool	Computes a line given two points from two vision tools.
	Sensor Window Tool	Detects motion in a window region. Ensures the scene is stable prior to other operations or waits until motion is detected. Can be used in place of physical sensors in flexible parts feeding where parts can take time to settle.
N/A	**Tool Filter Tool**	Takes the output from another tool that generates multiple sets of results, and returns a subset of the results based upon specified criteria.
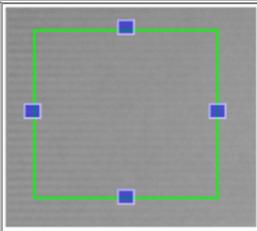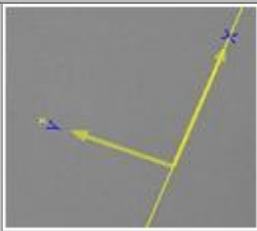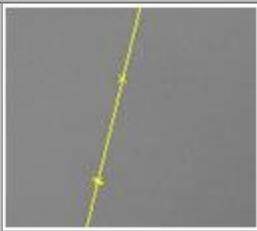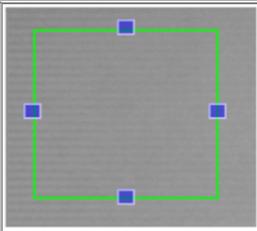
Graphical User Interface Overview

This section summarizes the purpose and use of each of the major components of the PreciseVision graphical user interface (GUI). The user interface has been designed to allow you to quickly construct a Vision Process and to easily define the properties for each of the utilized vision tools.

For detailed information on each of the components of the GUI, please read the "PreciseVision User Interface" chapter.
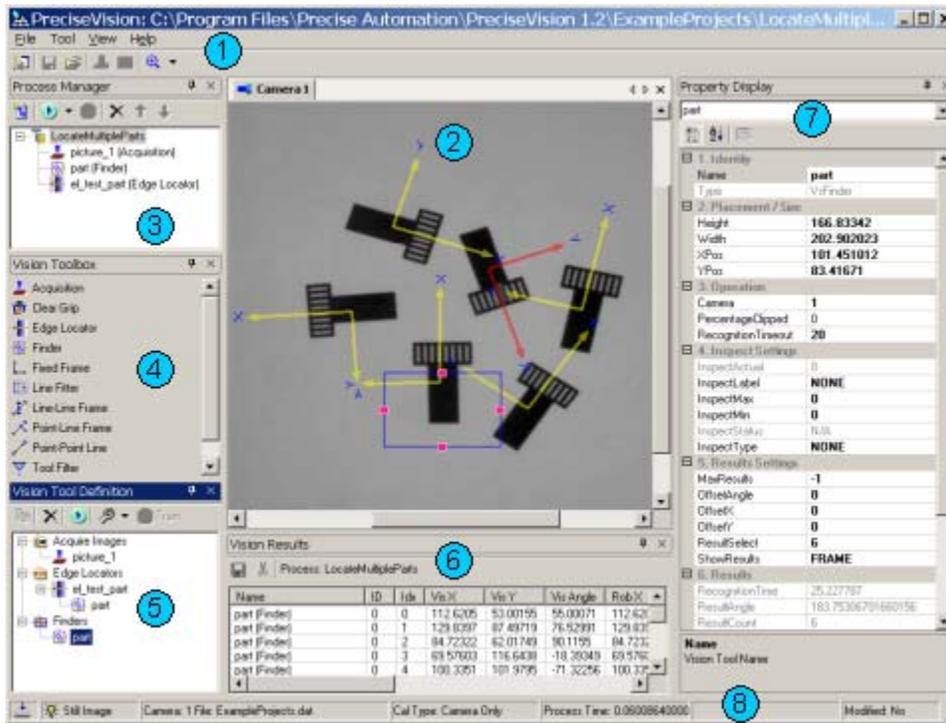
GUI Component	Description
1. Main Menu and Toolbar	Provides access to the major functions available within PreciseVision, e.g. loading and saving projects, selecting the camera output to view, etc. The toolbar includes buttons for putting the camera into "live video" mode, taking a single snap shot, and zooming the camera image up.
2. Vision Camera Display	Displays the output of the last selected camera. This can be either a frozen snapshot or a live camera image. When a Vision Process is being evaluated, the tool Results graphics will be overlayed on top of the camera image. When a tool is being edited, its outline will be displayed on top of the camera image. To reposition or resize a tool, simply drag its handles in this window.
3. Process Manager	Displays and permits editing of all of the Vision Processes defined within the currently loaded Vision Project. Each process can be expanded to show the sequence of Vision Tools that are executed within the process. Tools can be deleted or re-ordered within the process. To add a new tool, simply drag a defined tool from the Vision Tool Definition window and drop it on a process name. To preview how a Vision Process will perform, click on the "Test

	Process" green button in the toolbar. This will execute a process once or continuously without requiring a GPL procedure to trigger this activity.
4. Vision Toolbox	Lists all of the possible types of Vision Tools that can be created. To create a new copy of a tool, double click on the tool or right-click and select "Create tool". Each tool must be assigned a unique name. New copies of tools are automatically added to the lists in the Vision Tool Definition window.
5. Vision Tool Definition Window	Displays all of the tool that have been created. A created tool will not be utilized in a process until you drag the tool and drop it into the Process Manager window. If a tool is relative to another tool or requires other tools as inputs, they will be listed below the tool. This provides a quick visual queue as to the relationships between tools.
6. Vision Results Window	Whenever a Vision Process is executed, the output of each tool (i.e. its Vision Results) will be displayed in this window.
7. Property Display	Whenever a copy of a vision tool is selected in the Vision Tool Definition window, all of the properties of the tool will be displayed in this window. Dimmed properties are read only.
8. Status Bar	Displays status information such as the last Process execution time, live video/snapshot indicator, networking indicator, camera calibration type, etc.

Tutorial 1: Testing Processes and Editing Vision Tools

The first step in learning how to apply PreciseVision is to gain more of an understanding of how vision tools operate and how to edit tools. Fortunately, PreciseVision provides a very convenient means for experimenting with the tools without needing a camera, robot or any parts.

In the following procedure, you will be using a Sample Vision Project and camera images from disk files that are distributed with PreciseVision. One thing to note is that this Sample Project and image files were generated using standard PreciseVision facilities. So, once you are familiar with the system, you will be able to create your own Sample Projects and stored images to illustrate your work to co-workers or customers or to relay problems to Precise and our partners.

Prior to executing this tutorial, the following steps must first be performed:

- The procedure for "Installing PreciseVision on a PC" must be completed.
- No camera hardware of any sort is required.

In this first example, two Edge Locators are executed to compute a line along the side of a part.

> » Start by launching PreciseVision.   Click on ***"Start > Programs > Precise Automation > PV xxx > PreciseVision".***

If this is the first time PreciseVision is executed, a pop-up window will be displayed requesting the specification of the camera hardware followed by a window requesting that the software be activated. Both of these actions are good to perform and you should return to the installation chapter and follow the instructions for completing these procedures. However, if you wish, you can cancel out of these pop-up's since neither of these setup steps is required to initially execute the Sample Projects.

The following "Quick Start" pop-up window should now be displayed.

If you hover the cursor over the three buttons on the left, the text in the center window panel will change to provide more information about each selection.

> » Click on the *"Samples"* button. A standard file selection pop-up window will be displayed that lists a number of Sample PreciseVision Project files (*.pvp).
>
> » Click on the *"ComputeLineExample.pvp"* file and press *"Open"* to load this Project.

The default PreciseVision application layout should now be visible and the Camera Display should show the picture of a "T" part with two green Edge Locators as shown below.
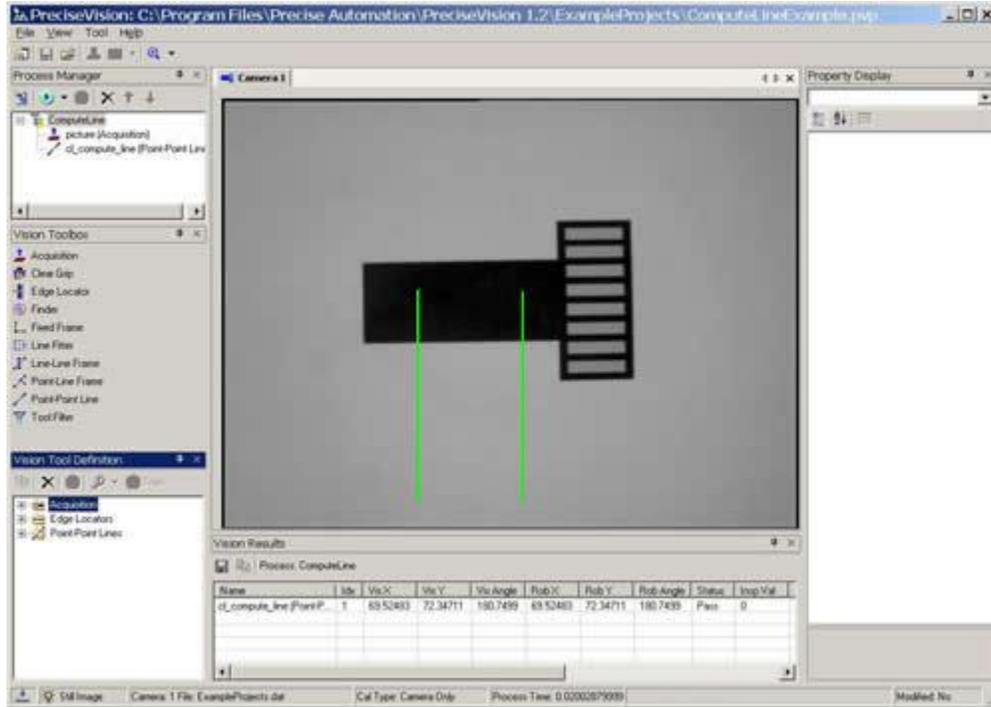
If you examine the Process Manager, you will see that this Project has one Vision Process defined, "ComputeLine". This Process consists of an Acquisition Tool, "picture", and a Point-Point Line Tool, "cl_compute_line". The Acquisition Tool would normally be set to capture an image from a camera. In this case, it is configured to read the image from a disk file. cl_compute_line does the work of computing a line from the output of two Edge Locators. The two Edge Locators are implicitly executed and not listed in the Vision Process (more about this later).

> » To test the Vision Process, click on **"ComputeLIne"** in the Process Manager and then click
> on the circular green **"Test Selected Process"** button in the window's toolbar.

Each time that you click on the "Test Selected Process" button, the Acquisition Tool loads an image file from the disk and then the Point-Point Line Tool executes the two Edge Locators and fits a line between their first detected edges. The first time you click the test button, the Camera Display should resemble the following. You will note that the output line of the Point-Point Line Tool is indicated in yellow.
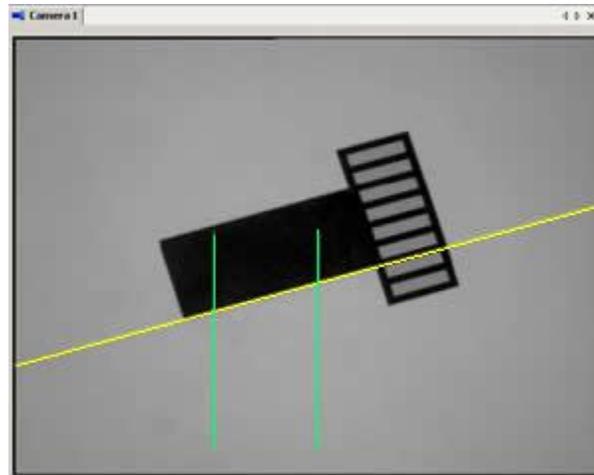
For this Sample Project, there are multiple image files and so the captured image and the results will change each time you click the test button.

> » To continuously execute the Sample Project, pull down on the arrow next to the test button in the toolbar and select *"Test Loop"*.
>
> » To stop the execution, click the *"Stop Test Loop"* button in the toolbar.
>
> » Next, to examine the defined tools, in the Vision Tool Definition Window, click on each of the *"+"* boxes to expand all of the displayed items.
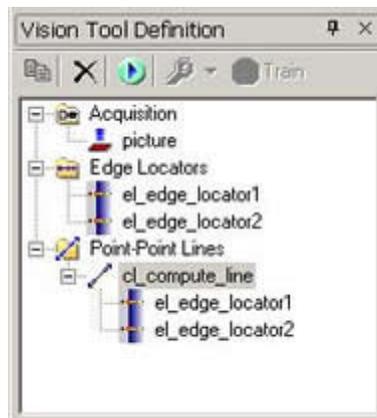
The items in the left-most column of the Vision Tool Definition Window are folders that contain all created tools of a specific type. If a type of tool has not been created, the corresponding tool folder is hidden. The top-level items listed in the tool type folders are the names of the created tools. The third and successive levels are lists of tools that are referenced by the tool above it. In this example, in addition to the "picture" and the "cl_compute_line" tools, two Edge Locators, "el_edge_locator1" and "el_edge_locator2" have been created. Note that the names of the Edge Locators appear twice, once in the top-level Edge Locator tool type list and a second time under the "cl_compute_line" tool. This second listing indicates that the output (i.e. "Results") of these two Edge Locators are referenced by and used in the computation of "cl_compute_line".

Since the system knows that the Edge Locators are needed for the computation of the Point-Point Line Tool (more about how to specify this later), the Edge Locators are automatically executed by the Point-

Point Line Tool if necessary and they need not be included in the Vision Process. Including the Edge Locators in the Vision Process before the computation of the Point-Point Line Tool is a matter of personal preference and does not effect execution time.

The Vision Tool Definition Window provides a convenient means to access all created tools whether or not the tools are included in any Vision Process. This window also provides a simple graphical representation of how tools are dependent on and connected to other tools.

> » In the Vision Tool Definition Window, click on either of the *"el_edge_locator1"* names to select this tool for editing.

Once the Edge Locator is selected for editing, it will be displayed in the Camera Display with graphical handles (as shown below) that allow it to be repositioned and resized.
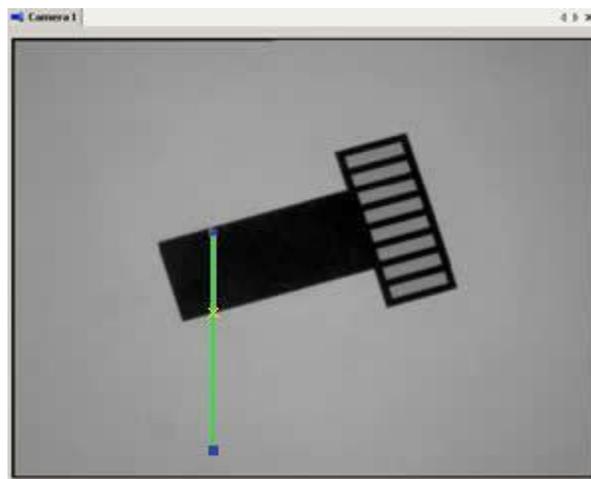
> » To practice repositioning the tool, click anywhere along the green line and drag the tool to translate it.
>
> » Click on the blue square that indicates the start of the Edge Locator and drag it to stretch or shrink the tool.
>
> » Click on the blue circle that indicates the end of the tool and drag it to rotate the tool's orientation.
>
> » To test the Vision Process with a new tool position and size, in the Process Manager, click on *"ComputeLIne"* and then click on the circular green *"Test Selected Process"* button in the window's toolbar.

When the Edge Locator was selected for editing, you might have noticed that its properties were automatically displayed in the Property List Display as shown below. In addition to graphically positioning and sizing this tool, you can also enter new values into this panel. The properties are organized into groups based upon their function. Readonly values are displayed in dimmed characters.

For each of the tool types, all of the properties are described in detail in the Vision Toolkit Chapter of this document.
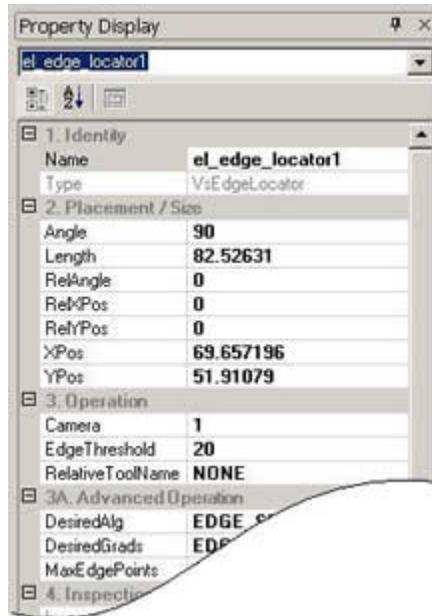
> » In the Property List Display, try editing the **"Angle"**, **"Length"**, and **"XPos"** and **"YPos"** properties and note the effect in the Camera Display.
>
> » If you wish to save your changes to the Sample Project, select **"File > Save Vision Project As..."** and assign a new name to your project so the original Sample is preserved.

This completes the tutorial on testing a Sample Project and editing a tool.

Tutorial 2: Adding Vision Tools

In this tutorial, we again make use of the **ComputeLineExample** Sample Project. In this case, we present the procedure for adding a Vision Tool to an existing Vision Process. For demonstration purposes, we will add an Edge Locator that measures the width of the stem of the "T" and validates if this dimension falls within an acceptable range. In an application that required high precision, you would probably use two Line Fitters for locating the edges of the "T" stem, but for training purposes, an Edge Locator will suffice.

Prior to executing this tutorial, the following steps must first be performed:

- The procedure for "Installing PreciseVision on a PC" must be completed.
- No camera hardware of any sort is required.

> » If PreciseVision is not running, click on **"Start > Programs > Precise Automation > PV xxx > PreciseVision"**.
>
> » If PreciseVision is already running, select **"File > Quick Start"**.

PreciseVision

If a pop-up window is displayed requesting the specification of the camera hardware or a window is displayed requesting that the software be activated, you can cancel out of these pop-up's since neither of these setup steps is required to initially execute the Sample Projects.

> » In the Quick Start window, click on the **"Samples"** button. A standard file selection pop-up window will be displayed that lists a number of Sample PreciseVision Project files (*.pvp).
>
> » Click on the **"ComputeLineExample.pvp"** file and press **"Open"** to load this Project.

The default PreciseVision application layout should now be visible and the Camera Display should show the picture of a "T" part with two green Edge Locators as shown below.

For robustness, the visual inspection should be designed such that the measured width of the stem is accurate independent of the orientation of the "T". To achieve this, the new Edge Locator should be oriented perpendicular to the computed Point-Point Line. If the Edge Locator was just placed at a fixed position and orientation, the width measurement would return too large a value if the "T" is rotated. To create the new tool, the following should be executed.

> » Double click on the **"Edge Locator"** in the Vision Toolbox.
>
> » In the Name pop-up, enter **"el_measure_width"** as the name of this new tool and press **"Ok"**. The new tool will be created and will appear in the Vision Tool Definition Window.
>
> » Click on the new tool in the Vision Tool Definition Window and drag it to the Process Manager and drop it on **"ComputeLine"**. This will evaluate the new tool after the Results of cl_compute_line are computed.
>
> » In the Property List Display, pull down the entries for **"RelativeToolName"** and select **"cl_compute_line"**. This makes the position and orientation of the new tool relative to the

28

computed position and orientation of the Point-Point Line Tool.

» In the Camera Display Window, resize and position the new tool so that it is approximately perpendicular to the yellow line for cl_compute_line and stretched across the stem of the "T".

The Camera Display should resemble the following picture that shows the el_measure_width tool drawn across the stem of the "T" and approximately perpendicular to the yellow line for cl_compute_line.
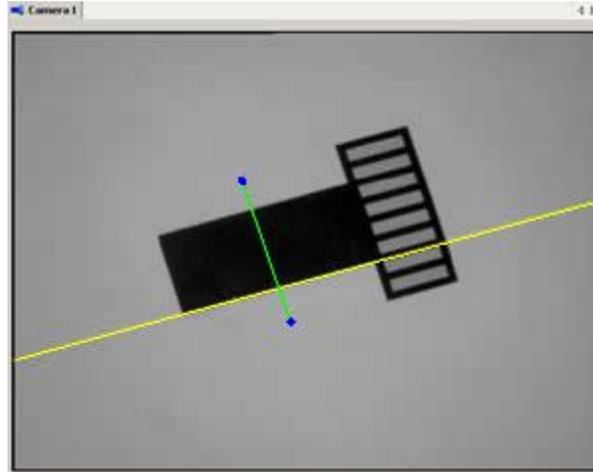
» To ensure that el_measure_width is exactly perpendicular to cl_compute_line, in the Property display, edit *"RelAngle"* to be an even multiple of 90. For example, if it has a value of -628.2, change this to -630.

» To automatically test the width of the stem, change *"InspectType"* to *"Distance_From_First_To_Last"*. This defines the inspected results as the computed distance between the first and last edge found by el_measure_width.

» To define the width dimensions that we will accept as valid, set the *"InspectMax"* value to *"31.8"* and the *"InspectMin"* value to *"31.2"*.

» To display a pass or fail label on the tool in the Camera Display, change *"InspectLabel"* to *"Pass_and_Fail"*.

» To test the new tool, in the Process Manager, click on *"ComputeLine"* to select it and then click on the circular green *"Test Selected Process"* button in the window's toolbar.

Each time you click the test button, you should now see the new tool in addition to the three former tools. The new tool should be drawn perpendicular to the yellow computed line and should be labeled to indicate if the width dimension is acceptable. The following illustrates a typical image in the Camera Display.
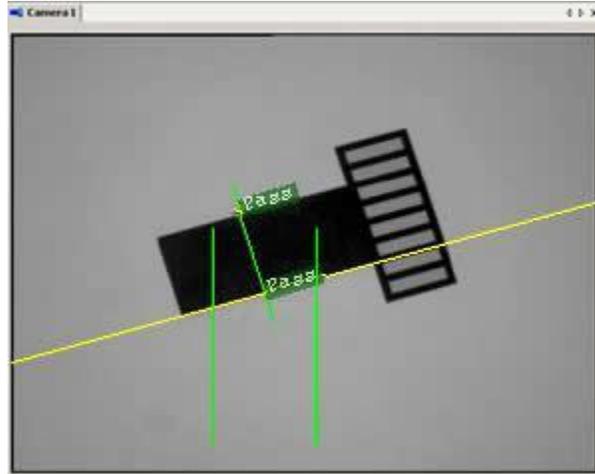
As you execute the modified Process, you should review the data displayed in the Results Window to see the numerical value of the computed width. All of the Results information is also available by selecting individual tools and viewing the data in the Properties List Display.

> » If you wish to save your changes to the Sample Project, select **"File > Save Vision Project As..."** and assign a new name to your project so the original Sample is preserved.

This completes the tutorial on adding a tool to an existing Vision Process.

Tutorial 3: Applying A Finder Tool

The Finder is the most general purpose tool available for identifying parts or sections of parts and locating their position, orientation and scale. For constrained or special situations, other tools can be combined to perform some of these functions. As an example, in the first tutorial, three tools were combined to locate a known part. But, in that case, the part was only expected to translate and rotate within a limited range. The Finder Tool utilizes an advanced patented algorithm to both identify and locate parts that are randomly positioned based upon a simple one-shot training technique.

In this tutorial, we step through the procedure for creating and teaching a Finder using a Sample Project. This tutorial also includes an example of creating a new Vision Process within an existing Vision Project.

Prior to executing this tutorial, the following steps must first be performed:

- The procedure for "Installing PreciseVision on a PC" must be completed.
- No camera hardware of any sort is required.

> » If PreciseVision is not running, click on **"Start > Programs > Precise Automation > PV xxx > PreciseVision".**
>
> » If PreciseVision is already running, select **"File > Quick Start".**

If a pop-up window is displayed requesting the specification of the camera hardware or a window is displayed requesting that the software be activated, you can cancel out of these pop-up's since neither of these setup steps is required to initially execute the Sample Projects.

> » In the Quick Start window, click on the **"Samples"** button. A standard file selection pop-up window will be displayed that lists a number of Sample PreciseVision Project files (*.pvp).
>
> » Click on the **"LocateRandomlyOrientedPart.pvp"** file and press **"Open"** to load this Project.

The default PreciseVision application layout should now be visible and the Camera Display should show the picture of a circular plastic part with a "V" shaped attachment as shown below.
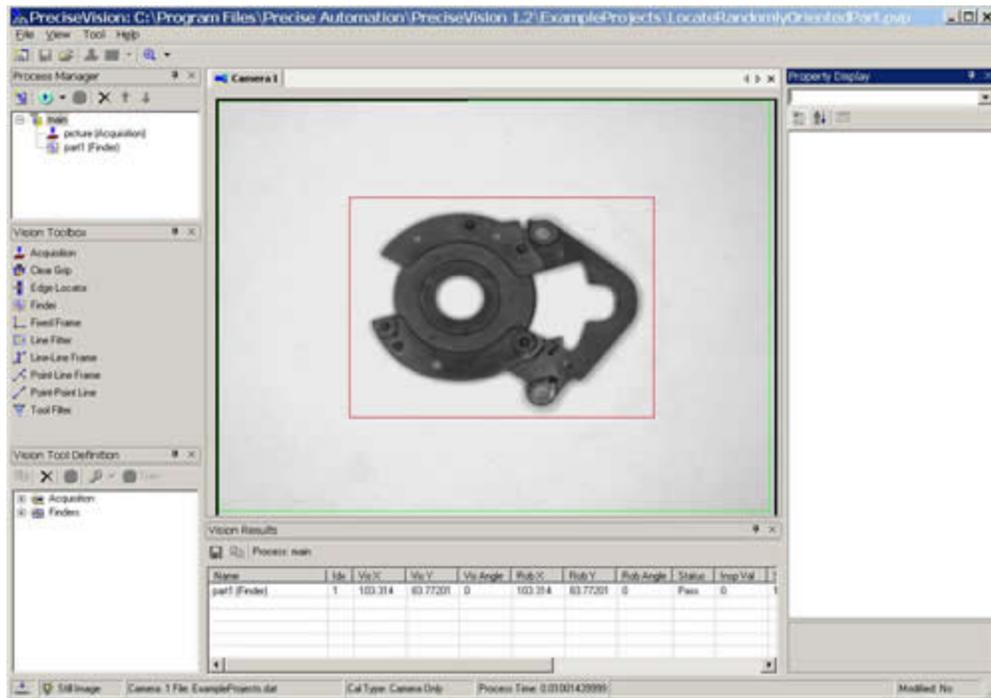
This Sample Project already includes a Finder as its principal tool. In this tutorial, we will go through the steps of re-creating this Vision Process including the Finder. Before beginning, the supplied Process should be tested to see how the Finder performs on this part.

> » To see the Finder in operation, click on **"Main"** in the Process Manager and then click on the circular green **"Test Selected Process"** button in the window's toolbar.
>
> » To continuously execute the Sample Project, pull down on the arrow next to the test button in the toolbar and select **"Test Loop"**.
>
> » To stop the execution, click the **"Stop Test Loop"** button in the toolbar.

We will now begin creating a new Vision Process. In order to perform the image capture, we will reuse the existing Acquisition Tool since it has already been initialized to read the desired image disk files.

> » Click on the **"Add a new Process"** button in the Process Manager toolbar.
>
> » In the Process Name pop-up window, fill in **"my_process"** as the name and press **"Ok"** to create the Process.
>
> » In the Vision Tool Definition Window, in the Acquisition folder, click on the **"picture"** tool and drag and drop it on top of the **"my_process"** in the Process Manager.

Before performing the next step of creating and training a new Finder, the following is background information and hints that will aid in applying this tool.

When you create a new Finder or re-teach ("re-train") an existing Finder, red and yellow rectangles will be drawn in the Camera Display as shown below.
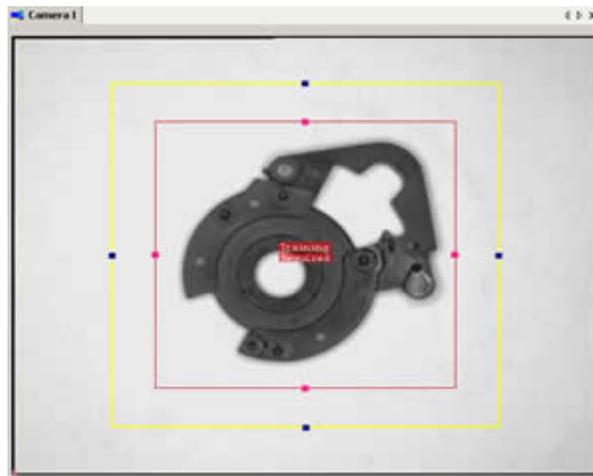
The red rectangle defines the "Finder Template" for the part to be located. When PreciseVision subsequently searches for the part, it is looking for matches for the Template in the image. When the system reports the position and orientation of a matched part, it is actually stating the position of the center of the Template when the match was found and the amount by which the Template was rotated. When defining the Template, keep in mind the following guidelines.

- The Template should fully enclose the target object and should contain no other objects or parts of objects.
- The Template should include some of the background on all sides of the target object, typically at least 16 camera pixels (too many is better than too few). This allows the system to distinguish the boundary of the object from the background, which is especially important for parts with few internal features. For a 640x480 camera, 16 pixels is 1/30th of the full height of the camera image.
- If the background varies considerably (i.e. it is dirty or has blotches), the part should be trained on a uniform neutral background.
- The target object should be oriented in the camera image in what you want to define as 'Zero' degrees. That is, when the Finder later locates a part is in the trained orientation, it will return an orientation angle of 0 degrees.

The yellow outline defines the "Training rectangle". This rectangle is used to teach the system how to distinguish the Template from other objects or background clutter. When defining this rectangle, keep in mind the following guidelines.

- This rectangle must be larger than the Template and must fully enclose the Template.
- This rectangle should include one and only one complete copy of the target object. It's okay if this region includes other partial copies of the target object so long as each partial copy is less than 50% of the target object.
- This rectangle should include any other objects or clutter that are expected to be encountered during runtime to aid the system in learning to distinguish between the target object and its surroundings.
- This rectangle should be as large as possible without violating the rules above. A larger region permits the system to test more positions and orientations for the Template during the training phase and assists in optimizing the runtime search strategy.

It's now time to create the Finder and add it to the Vision Process.

> » Double click on the *"Finder"* in the Vision Toolbox.
>
> » In the Name pop-up, enter *"my_finder"* as the name of this new tool and press *"Ok"*. The new tool will be created and will appear in the Vision Tool Definition Window.
>
> » Follow the step-by-step instructions within the "Finder training" Application Wizard that is displayed to setup the Finder keeping in mind the hints above.
>
> » Click on *"my_finder"* in the Vision Tool Definition Window and drag it to the Process Manager and drop it on *"my_process"*. This will evaluate the Finder after the camera acquisition is executed.
>
> » To test the new tool, in the Process Manager, click on *"my_process"* to select it and then click on the circular green *"Test Selected Process"* button in the window's toolbar.

Congratulations, you just created your first Vision Process that identifies and locates randomly parts! You can continue to click on the test button or execute your process in a continuous loop.

> » If you wish to retrain the Template at anytime, click on *"my_finder"* in the Vision Tool Definition Window to select it. Then click on the arrow to the right of *"Finder Operation Mode"* icon on the window toolbar and select *"Train Mode"*. You can also restart the Finder Training Wizard in this same pull-down if you wish.
>
> » If you wish to save your changes to the Sample Project, select *"File > Save Vision Project As..."* and assign a new file name to your project so the original Sample is preserved.
>
> » At this time, we highly recommend that you execute each of the distributed Sample Vision Projects, review their Vision Processes and edit the Tool Properties to get an understanding of more of the vision tools.

This completes the tutorial on creating a Vision Process and applying the Finder Tool.

PreciseVision

Tutorial 4: Developing a Robot Vision Application

This tutorial illustrates how to develop a simple, but complete, robot vision guidance application. It presents both the PreciseVision Project and the GPL software that must be executed on a Guidance Controller. This tutorial requires that your robot and Guidance controller be operational and interfaced to the PC executing PreciseVision.

Prior to executing this tutorial, the following steps must first be performed:

- The procedure for "Installing PreciseVision on a PC" must be completed.
- A camera must be rigidly mounted over a region that can be reached by the robot.
- The Ethernet camera interface must be installed and interfaced to PreciseVision using one of the procedures in the "Hardware Installation" section of this manual.
- The robot must be fully operational and interfaced to the PC via Ethernet.
- The procedures outlined in "Configuring Communication Between the PC and the Controller" that permit the two systems to exchange data must have been performed.
- The "Robot Vision Area Camera Calibration" must have been executed to define the camera's pixel-to-millimeter conversion and the transformation from the camera's frame of reference to the robot's frame of reference.
- The Guidance Development Suite must be installed on the PC and you should have a modest ability to develop and execute GPL programs.

» If PreciseVision is not running, click on **"Start > Programs > Precise Automation > PV xxx > PreciseVision".**

» If PreciseVision is already running, select **"File > Quick Start".**

» In the Quick Start window, click on the **"New Project"** button.

You should now see the "Create New Project From Application Template" pop-up window that resembles the following.
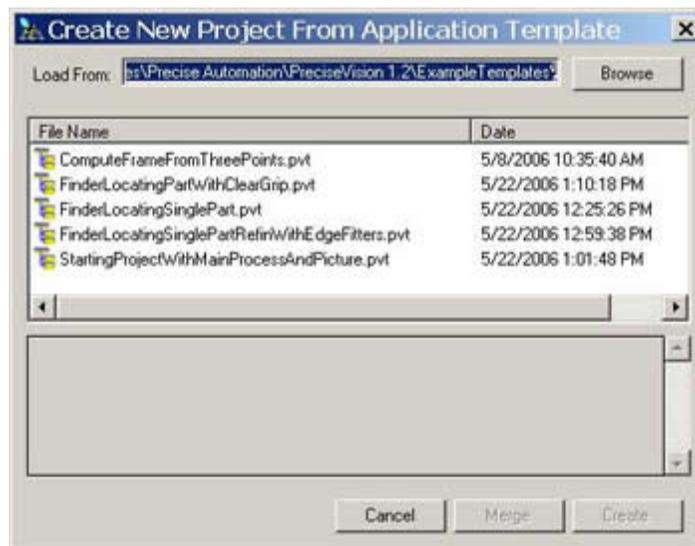
The "Create New Project From Application Template" is a wizard that builds a complete Vision Project for a number of common tasks. For example, the "FinderLocatingPartWithClearGrip" constructs a Project that uses a Finder Tool to locate an object and then executes two ClearGrip Tools to verify that the parts can be grasped without hitting an obstruction. In particular, this wizard deletes the current Vision Project (after confirmation if the Project should be saved), creates a new Project, and automatically creates the Vision Tools and Vision Processes for the selected task. The wizard then walks you through the steps to position, size and train each of the tools for your parts. Over time, we will be adding additional templates to address even more common tasks and we will be adding the ability for you to add your own templates to simplify the use of the system by your co-workers or customers. Of course, you can always create your own Vision Projects, Processes and Tools using the basic methods described in the other tutorials. The Application Wizard is just a convenience for people learning about the system or it can serve as a starting point for those already familiar with the system.

For this exercise, we will be selecting the "FinderLocatingSinglePart" template. This takes a single picture and then employs a single Finder Tool to locate your part.

> » Click on the *"FinderLocatingSinglePart.pvt"* file, click on *"Create"* and then follow all of the instructions of the wizard to create your Vision Project. At the conclusion of the wizard, the Project, Process and Tools should all be created and the Finder should be trained on your object.
>
> » Save your Project by selecting *"File > Save Vision Project As..."* and assigning a file for your Project.
>
> » To test your Project, place your object in the camera's field-of-view, click on *"Main"* in the Process Manager and then click on the circular green *"Test Selected Process"* button in the window's toolbar.
>
> » To validate that the Vision Project and the camera calibration are working properly, using the same robot pointer you used during the camera calibration process, manually move the robot to the X/Y position indicated in the Results Window *"RobX"* and *"RobY"* values and verify that the tool is at the center of your part's Finder Template.

PreciseVision is now setup and ready to receive commands. It is not necessary to place PreciseVision in a special mode to work with the robot, it is always listening for commands from a GPL system. So, there is no further development to be done in PreciseVision.

The next step is to develop a GPL project that will execute the Vision Process "Main", retrieve the "part1" Finder Results and move the robot to the specified position. A simplified version of the required program is presented in the *"Guidance Programming Language, Introduction to GPL"* manual in the Vision section and is repeated below.

> » Within GDE, the Guidance Development Environment, in the Project Manager, create a new GPL Project named *"Robot_vision"*.
>
> » In the new *"Robot_vision"* Project, double click on the *"Main.gpl"* file to begin editing.
>
> » In the editor window, below the "Public Sub Main" statement, insert the following lines of text. If you are reading this exercise in the *Precise Documentation Library* (the online help file), this can be accomplished by copying and pasting if you wish to save yourself some typing.

```
        Dim vis As New Vision
        Dim vResult As New VisResult
```

```
        Robot.Attached = 1
        Move.Loc(safe, vsProfile)

        vis.Process("Main") ' Run vision process "Main"
        If vis.ResultCount("part1") = 0 Then
            Console.Writeline("Vision object not found")
            Goto done
        End If
        vResult = vis.Result("part1", 1) ' Get results

        ' Create a reference frame object and set it
        ' equal to the returned vision location
        Dim vsRefFrame As New RefFrame
        vsRefFrame.Loc.PosWrtRef = vResult.Loc

        ' Pickup point is relative to new frame
        vsRelPoint.RefFrame = vsRefFrame

        Move.Approach(vsRelPoint, vsProfile)
        Move.Loc(vsRelPoint, vsProfile)
        Move.Approach(vsRelPoint, vsProfile)

        ' Move back to safe location
        Move.Loc(safe, vsProfile)

    done:
```

» In the the *"Robot_vision"* Project, double click on the *"GModule.gpo"* file to access the global Motion Objects panel.

» Create and teach a Location named *"safe"* that positions the robot so it is not obstructing the camera's view of the part. Also, create a Motion Profile named *"vsProfile"* that will be used to slowly move the robot to the part.

» Create a Location named *"vsRelPoint"*. This will define the robot's pickup position relative to the Location returned from PreciseVision. Normally, it would be best to manually define this Location relative to the actual value returned from the vision system. But for demonstration purposes, set its properties to the following where the Z value is a safe world coordinate value above the part.

X = 0
Y = 0
Z = Z_safe
Yaw = 0
Pitch = 180
Roll = 0

» Press the *"Save Document"* icon on the main tool bar to save your changes.

» To load your application, in the Project Manager, drag the *"Robot_vision"* Project from the PC to the top *"Loaded Projects"* panel and drop it.

You have now completed the development of the GPL Project and we are ready to execute the complete vision guided robot application.

> ⚠️ **DANGER**: Before proceeding with this procedure, please ensure that the robot has been properly mounted, all required safety interlocks have been installed and tested, and power has been connected. For the PrecisePlace robots, this information is provided in the *"PrecisePlace 2300/2400 Robot, Hardware Introduction and Reference Manual"*.

» In the GDE Robot Control Window, set the **"Robot Speed"** to approximately 5%. This will force your Project to move the robot at 1/20th of its normal speed.

» **This next action will move the robot automatically to a position computed by PreciseVision, so be prepared to hit the E-Stop button or disable power if any problem occurs!**

» With your part placed in the camera's field-of-view, execute the GPL Project. When you run a GPL program for the first time, it is advisable to start the project in debug mode. This can be done by selecting **"Debug > Start with break"** or pressing **"Shift-F5"**. Single step through the program one line at a time by pressing **"F10"**.

Congratulations, you just created your first full vision guided robot application!

For more information on the GPL vision interface, please read the Vision Guidance chapter of the *"Guidance Programming Language, Introduction to GPL"* manual. This contains an overview of the built-in GPL vision classes, methods and properties. For more detailed information, please see the *"Guidance Programming Language, GPL Dictionary Pages"*.

Graphical User Interface

PreciseVision User Interface

The PreciseVision Graphical User Interface (GUI) consists of a title bar that displays the name of the opened project, a top menu bar, a main toolbar, and a variety of dockable windows. Each of the dockable windows can be displayed or hidden, resized and repositioned into the arrangement that is most efficient for your use.

When PreciseVision is started for the first time, the following default arrangement of the various elements of the GUI is presented.
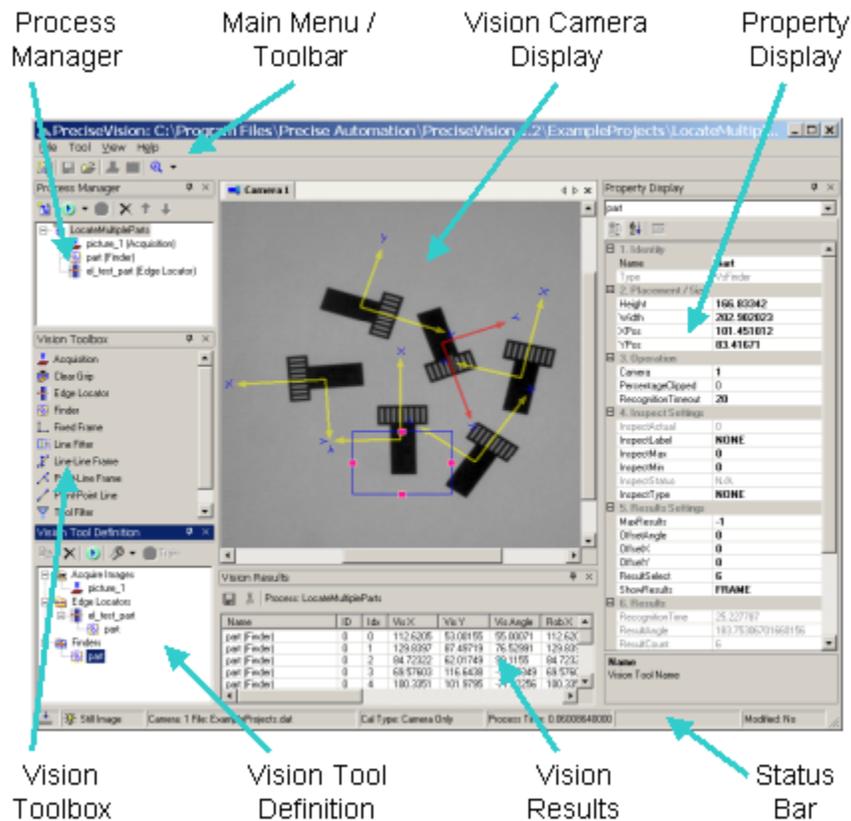
To reposition a window, you simply click in its title bar and drag it to its new location. If you drag a window on to another window, they can split the space or share the space using tab controls. If you click on the close icon (x) in the top right of a window's title bar, you can use the "View" top-level menu to redisplay the window. If you click on the "push pin" in the top right of a window's title bar, you will either "pin" a window and fix its location or "un-pin" a window so it can share its space with another window.

Windows can be resized by grabbing a border and stretching or shrinking it to the desired dimension.

In the following sections, the functions supported by each of the major elements of the PreciseVision GUI (i.e. main menu bar, tool bars, and windows) will be described. In addition, the Preferences Panel for specifying system configuration information is also described. The graphical components are presented in order of importance rather than alphabetically. To jump to a specific description, click on a label in the picture above or on a window.

Main Menu and Toolbar

The Main Menu and the Main Toolbar provide access to general functions such as displaying the dockable windows, saving and loading Vision Projects and launching the camera calibration operations. This menu and toolbar appear as follows.

The following tables describe the operation of each of the selections within the Main Menu and are organized by the name of the top-level menu. These tables are followed by a description of each of the Toolbar buttons.

File	Description
Quick Start	Displays the pop-up Quick Start window that is shown each time PreciseVision is started. This window allows an existing Project to be loaded, sample canned Projects to be loaded, or a new Project to be constructed by a step-by-step wizard that supports many common procedures. The latter two options are very instructive for individuals who wish to learn about machine vision or PreciseVision.
New Vision Project	Displays a sub-menu with selections that create either an empty un-named Vision Project or a new Project constructed by a step-by-step wizard that supports many common procedures. An additional sub-menu selection permits the last executed wizard to be re-executed to allow editing of the previously generated Vision Process.
Open Vision Project	Opens a new Vision Project from the disk and loads it into memory replacing the current Vision Project. The standard file pop-up window is displayed to allow browsing to the desired folder. PreciseVision project files have a ".PVP" file extension.
Save Vision Project Save Vision Project As ...	Either saves the loaded Vision Project to its associated disk file or displays a pop-up file window to allow the operator to browse to the desired disk folder and assign a disk file name.
Close Project	Closes the loaded Vision Project after prompting if any changes should be saved to the Project's disk file. The close operation deletes all defined tools and Vision Processes resulting in an empty un-titled Project.
Preferences	Displays the PreciseVision Preferences panel. This panel contains a number of application setup parameters that customize the operation and appearance of PreciseVision, e.g. enable/disable the display of the Main Toolbar or tool graphics.
Acquire New Image	Takes a single snap shot from the currently selected camera and loads the image into the Camera Display window.
Image Load / Save	Diagnostic and debugging aid that loads a camera image from a disk

	file into the Camera Display window or saves the currently displayed image to a disk file.
Exit	Terminates execution of PreciseVision

View	**Description**
Layouts	Provides "Load Default", "Load Layout" and "Save Layout" selections for permitting custom desktop layouts to be preserved, reloaded or set back to the normal default. Also provides "Load Runtime Layout", "Load Edit Layout" and "Load Calibration Layout" for putting standard layouts into effect. These standard layouts can also be accessed from the main toolbar.
Select Camera	Selects the camera whose image is to be presented in the Camera Display window. This also specifies the camera that is utilized in the Camera Calibration methods and other functions.
Debug	Provides access to a sub-menu that launches a number of diagnostic functions.
Process Manager	
Vision Tool Definition	Permits any dockable window that has been closed to be re-displayed.
Vision Toolbox	
Property Display	
Results Display	
Guidance Web Panels	Opens up to three web panels specified in the 'Preferences -> Web Interface' tab. This provides convenient access to the Precise Guidance Controller web interface.

Tool	**Description**
Calibrate Vision	Initiates the execution of the camera calibration wizard. This procedure determines the pixel-to-millimeter conversion for the selected camera and also the conversion from the camera's frame of reference to that of the robot.
Camera Calibration Configuration	Displays a pop-up window that associates a camera calibration file with a specific camera in the current Vision Project. The information in this pop-up is saved and reloaded with the Vision Project.
Show Camera Information	Displays a pop-up window that shows which camera is selected and the camera calibration data that is in effect.
Camera Gain and Offset	Displays a pop-up window that permits the brightness range and offset for the currently selected camera to be set. This is helpful during the setup procedures for the Camera Calibration methods and other functions.

Help	**Description**
Contents	Opens the *Precise Documentation Library* in a separate window and jumps to the specified section that is relevant to PreciseVision.
PreciseVision GUI	

Vision Tools Overview	
GPL Programming	
Product Activation	Displays the pop-up window used for activating this product.
Camera Driver Info	Presents the version and identification information for installed camera drivers.
About PreciseVision	Generates a popup window that displays the PreciseVision version and ID information along with build information for key components of this application package.

Icon	Tool Tip Title	Description
	Create new Vision Project	A new empty un-named Vision Project is created by deleting all of the tools and existing Vision Processes. If the current Project has been changed and needs to be saved to the disk, a pop-up is first presented that permits the operator to save the modifications.
	Save current Vision Project to disk	Saves the loaded Vision Project back into its disk file. If the current project has never been saved to the disk, a file pop-up window is displayed to allow the operator to browse to the desired disk folder and assign a disk file name.
	Open a Vision Project	Opens a new Vision Project from the disk and loads it into memory, replacing the currently loaded Project. The standard file pop-up window is displayed to allow browsing to the desired folder. PreciseVision project files have a ".PVP" file extension.
	Display live video camera image	Places the currently selected camera into "Live Video" mode. Pictures will be taken continuously and displayed in the Camera Display window. This mode is very convenient for setting up the focus, f-stop, gain, and offset of a camera. If the camera is already in live video mode, pressing this button will terminate this mode.
	Take a single snap shot	Takes a single snap shot (i.e. picture) using the currently selected camera and displays the results in the Camera Display window.
	Toggle Color Display Mode	For color cameras only, toggles the Camera Display window display between color and grayscale mode. Set to color display mode by default.
	Zoom the Camera Display window	Zooms the Camera Display window view in or out. Pressing the button zooms in. A pulldown menu tied to this button zooms out or resets the window to its un-zoomed state. The supported zoom factors are 0.25, 0.5, 1, 2, and 4.
	Switch display layout	Switches between three predefined display layouts: Edit, Run and Calibration. This icon makes changing the display layout very easy.

Vision Camera Display

The Camera Display window shows the captured image from a selected camera with graphics to illustrate applicable vision tools.   The camera image can be a "live video" display, which means that the captured image is continuously updated, or a single snap shot. The vision tool graphics illustrate the placement of tools and their computed results. When a tool is selected for editing, the graphics also include handles that permit the tool to be easily re-positioned, re-oriented and re-sized using a mouse.

The following picture is a typical camera image with overlayed tool graphics. For all cameras, the X-axis of the camera coordinate system points from left to right and the Y-axis points from the bottom of the display to the top. The origin (0,0) of the camera coordinate system is in the lower-left corner of the camera image. If the camera is not calibrated, X/Y positions in the camera image are returned in units of raw pixels relative to the lower-left corner of the camera image. If the camera is calibration, X/Y positions are typically in units of millimeters relative to the origin.
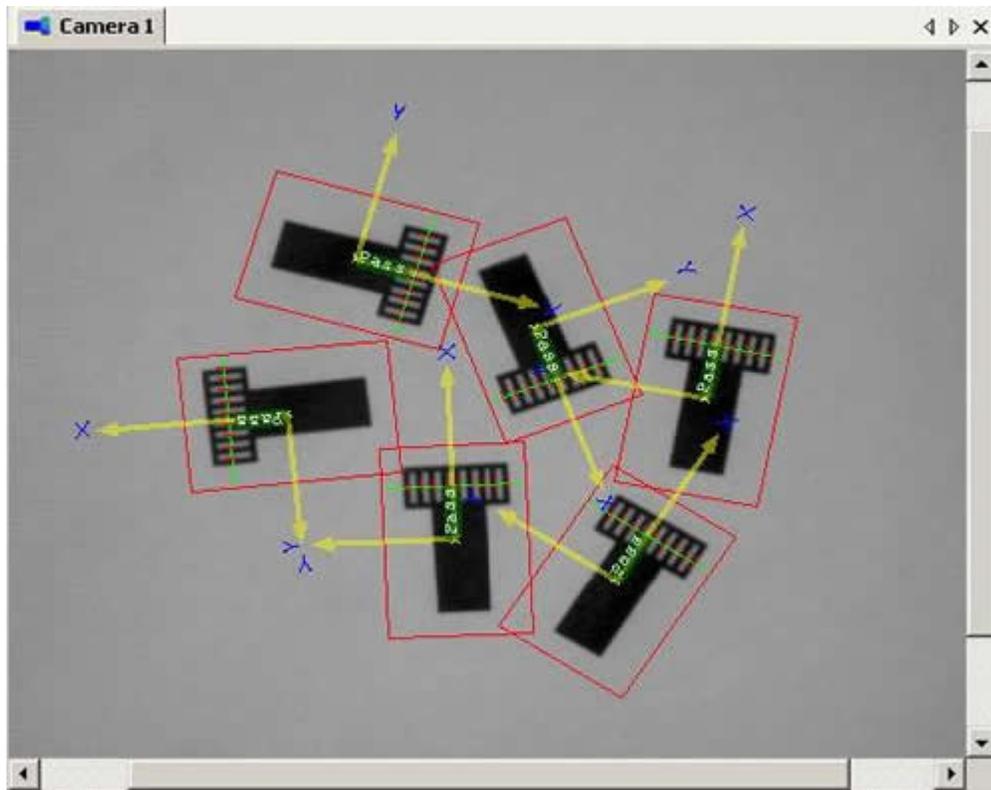
To accommodate large or small images (those with many or few pixels), the camera display can be zoomed out or zoomed in. The control for this operation is provided on the Main Toolbar along with icons for manually placing a camera into live video mode and taking a snapshot. The Main Menu contains other commands for changing the selected camera and adjusting their gain and offset.

The following images describe how to utilize the graphical handles to place and size a typical vision tool and options for controlling how the results of a tool are illustrated in the Camera Display Window.

Graphical Display	Description

	This is a snap shot of a Line Fitter Tool placed over one edge of a part. The green and blue perimeter lines indicate the tool's search area. The blue edge marks the side of the search region that should be positioned on the dark side of the edge. The red line illustrates the results of the tool and marks the best fit line. To position the tool, click anywhere in the search region and drag the rectangle. To resize the tool, click one of the four blue squares and drag. To rotate the search region, click the blue circle and drag.
	If the output of a tool is to be utilized as an input (X, Y) point for another tool, it is informative to display the result as a single position. In this picture, the **ShowResults** property for the Line Fitter has been set to "POINT". This places a yellow "X" at the origin of the output of the Line Fitter. Note, setting **ShowResults** only affects the graphical display and does not alter the results computed by the tool.
	In this example, if the output of the Line Fitter is to be interpreted as a line, the **ShowResults** property for the Line Fitter can be set to "LINE".
	In this final example, the output of the Line Fitter is to be interpreted as a reference frame, so the **ShowResults** property for the Line Fitter has been set to "FRAME".
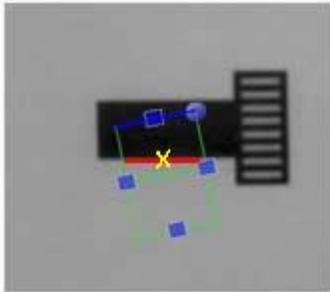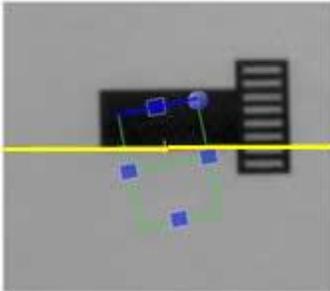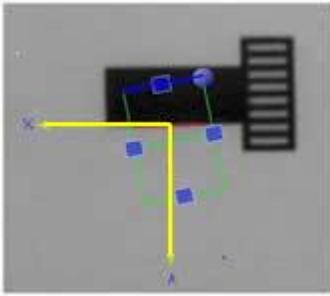
Process Manager Window

The Process Manager Window displays the list of all of the Vision Processes that are contained in the currently loaded Vision Project. If a Vision Process is expanded, all of the Vision Tools for the Process are displayed in the order in which they are evaluated. This window is the primary means for managing Vision Processes including their creation, deletion, modification and test execution. Any Process that is displayed in this window is eligible for execution by a GPL procedure running on a Precise Guidance Controller.

A typical Process Manager Window is shown below.
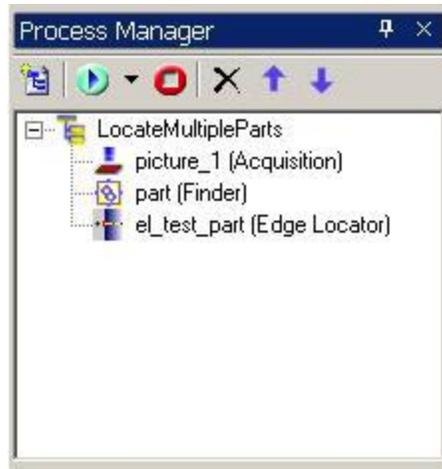
In this example, a single Vision Process "LocateMultipleParts" is contained in the currently loaded Vision Project. This Vision Process has three tools. The Process begins with the standard Acquisition Tool that takes a picture. After the picture is captured, the system evaluates a Finder Tool named "part". The final operation is an Edge Locator "el_test_part".

It should be noted that if the Edge Locator is defined relative to the Results of the Finder Tool, the Process could have been written without the explicit execution of the Finder. When the system encounters the Edge Locator, it would automatically execute the Finder if the Finder had not been evaluated. The decision of whether to include the Finder in the Vision Process is entirely a matter of personal preference. Whether or not the Finder is included, the Results of the Process would be identical as well as the execution time.

To add a new Vision Process to a Project, click the toolbar icon to "Add a new process". Once the new Process is assigned a unique name, it will appear as a top-level (i.e. left-most) item in the window. To specify the sequence of tools to be executed in the process, simply drag Vision Tools from the Vision Tool Definition Window and drop them on the Process. New tools are automatically added to the end of the Process and can be re-ordered by using the up and down arrows in the toolbar. The order of appearance of the tools is important since the tools are evaluated top to bottom in the displayed list.

Once a Vision Process has been created, it can be triggered from GPL or can be manually tested using an icon on the toolbar. The manual test can be executed a single time or the process can be executed continuously. The continuous test mode is really for demonstration purposes since the execution of the Process will not be synchronized with the GPL robot program and therefore not of very much practical use.

The following tables provide additional information on each of the window toolbar options.

Icon	Tool Tip Title	Description
	Add new process	Creates a new Vision Process and adds it to the Process Manager Window after it has been assigned a unique name via a pop-up window. The Process names are very important since they are used by GPL procedures to initiate the execution of a Process.
	Test selected process	Test executes the selected Process once or continuously. This is useful for debugging a Process and examining its computed results. However, in a robot guidance application, Processes are always

		initiated from a GPL procedure to ensure that the picture taking and evaluation are coordinated with the robot motions.
⬢	Stop test loop	If a process is running in a continuous test mode, this button will stop the Process. This is only used while manually test executing a Process.
✕	Remove selected item	Removes a Process from a Project or a tool from a Process. In both cases, any referenced tools are not deleted from the Project, only the reference to the tool is removed. To delete a tool, you must use the Vision Tool Definition Window.
⬆	Move tool up	Moves a vision tool up or down one line in a Vision Process list. The order in which the tools are listed in a Vision Process is important because this is the order in which the tools are evaluated.
⬇	Move tool down	

Vision Toolbox Window

The Vision Toolbox Window displays a list of all of the vision tool types that can be utilized to construct a Vision Process. Currently, the list of tool types appears as shown below although new tool types are in the process of being developed.
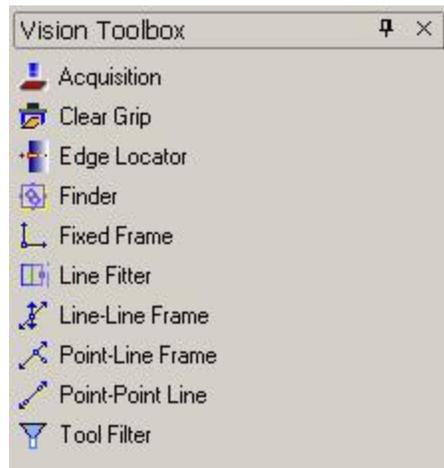
To add a new tool to an application, double click on a tool type or right-click and select **"Create Tool"**. A pop-up will be displayed for assigning a unique name to the new copy of the tool. Once a name has been assigned, the new tool will appear in the Vision Tool Definition Window. Once there, the tool can be edited or dragged into the Process Manager to add the execution of the tool to a specific Vision Process.

Vision Tool Definition Window

The Vision Tool Definition Window displays all of the tools that have been created in the currently loaded Vision Project, even those that are not referenced in any Vision Process. This window provides the primary means for adding tools to a Vision Process and for selecting tools to be edited. As a convenience, this window graphically indicates when tools are dependent upon the results of other tools.

PreciseVision

A typical Vision Tool Definition Window will appear as shown below.
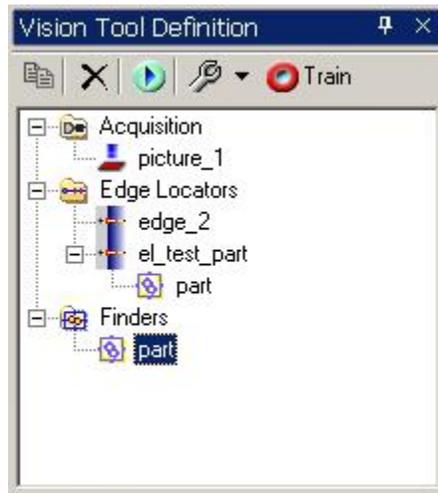
To simplify accessing tools, all of the defined tools are grouped according to their tool type and only tool types that have one or more tools defined are visible. In the example above, there is one Acquisition Tool named "picture_1", two Edge Locators named "edge_2" and "el_test_part" and one Finder Tool named "part". The second Edge Locator has been defined with respect to the results of the Finder, so the Finder "part" appears below "el_test_part" to indicate this relationship.

There are a number of ways that a new tool can be added to the Vision Project and the Vision Tool Definition Window:

-
    - Double click on a tool type in the Vision Toolbox Window.
    - Right-click a tool or tool type in the Vision Tool Definition Window and select "New Vision Tool".
    - Drag a tool type from the Vision Tool Definition Window and drop it into the Camera Display Window.
    - Right-click in the Camera Display Window and select "New Vision Tool".

To copy an existing tool while maintaining its dependencies to other tools, highlight the tool in this window and click the "Copy" icon in the toolbar or drag the existing tool and drop it into the Camera Display Window.

To edit a tool, simply click on the name of the tool. This will automatically display the properties of the tool in the Property Display Window and will highlight the editing graphics and the handles for the tool within the Camera Display Window.

To test the operation of a tool without executing a Vision Process, click on the tool and click the "Play" button in the toolbar. This only executes the selected tool and any tools on which it is dependent. If the tool on which it is dependent returns multiple results, e.g. a Finder, only the first set of results is processed by the tool being executed. This is done to permit you to easily reposition the executed tool.

The following tables provide additional information on each of the window toolbar options.

Icon	Tool Tip Title	Description
	Create copy	Copies the selected tool along with its relationships to

46

		other tools.
✕	Remove selected tool	Deletes the selected tool after confirmation but does not effect any tools that it references.
▶	Execute selected tool	Executes the selected tool and all of the tools on which it is dependent without executing a Vision Process. See above for further information.
🔧	Finder operation mode	For Finder Tools, switches between "Find" and "Train" mode. Can also launch the Training Wizard to assist in teaching a Finder Tool and can display a trained Finder template at various resolutions as a diagnostic aid.
●	Perform training operation	For tools that require training, executes the training process. In the case of the Finder Tool, this records the Finder Template and performs the analysis to determine distinctive features and the maximum levels of the search pyramid.

Property List Display

In PreciseVision, each Vision Tool is internally represented as a software "Object". The parameters for each tool are referred to as its "Properties". The properties include both the values that specify how a tool operates (e.g. its tool type, tool name, position, size, operating characteristics, etc.) as well as the values that indicate the results of executing the tool (e.g. number of objects found, number of edges located, etc.). For a given type of Vision Tool, the list of properties is always the same, but the list will vary from one type of tool to the next.

Some Vision Tools provide special buttons that are displayed between the tool name and the property grid. These buttons provide access to tool specific operations during the editing process (e.g. automatically snapping a tool into position).

The Property List Display provides a very convenient means for both viewing and editing the property values of a tool. As an example, the Property List Display for a typical Fixed Frame Tool is shown below.
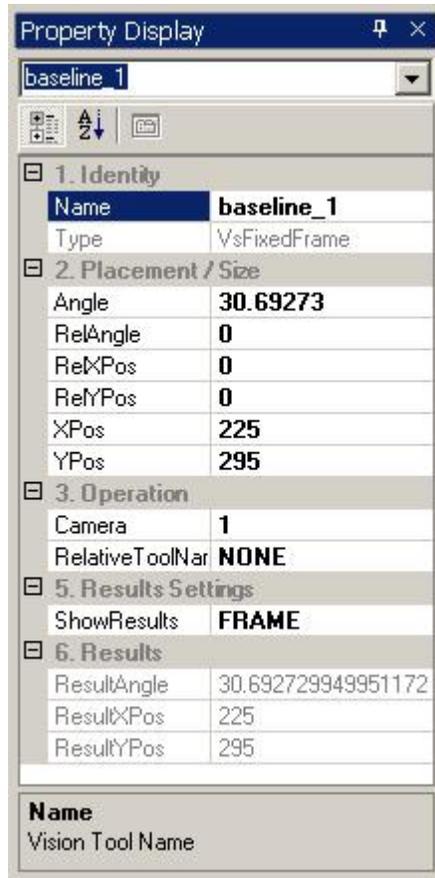
The pull-down list at the top of the window selects the tool to be displayed. The displayed tool will also be automatically changed whenever a new tool is created or a tool is selected in the Vision Tool Definition Window.

Once a tool has been selected, any of its writable values can be modified by clicking on a cell and entering a new value. When a property is selected, a short description of the property is displayed at the bottom of the window. Properties that are readonly are indicated by dimmed text.

To assist in accessing these parameters, the properties are group according to their use. A description of each of the groups is presented in the following table. For detailed information on the properties for each of the tool types please see the chapter on the "Vision ToolKit".

Property Group	Description
1. Identity Properties	Specifies the name of the tool and its basic tool type.
2. Placement / Size Properties	Defines the position, orientation and size of the tool in the coordinate system of the camera. In addition, if the tool is defined relative to another tool, the relative position and orientation can also be specified. Normally, it is easiest to graphically position and size each tool by dragging on the tool's handles displayed in the Camera Display Window.
3. Operation Properties	Defines parameters that control the execution of the tool. For example, a tool might have an Operation Property that defines if it should screen out black-to-white edges or the gray scale levels above which pixels are processed. Only the

	commonly used Operation Properties are placed in this group.
3A. Advanced Operation Properties	Contains Operation Properties that are not normally modified and generally require a bit more experience to utilize. For these properties, their default settings will yield good tool performance for typical applications.
4. Inspection Settings Properties	Permits a single scalar result of the vision tool to be automatically tested (inspected) to verify that it is within limits. A Pass / Fail property is set accordingly and a Pass / Fail label can be optionally displayed on the tool in the Camera Display Window.
5. Results Settings Properties	Controls the results that are computed by the vision tool. For example, if multiple sets of results can be computed by the tool, these properties can limit the number of sets of data computed and can select the set of data that is displayed in the **Results Properties**.
6. Results Properties	Displays a single set of results for the vision tool. In general, the results returned varies from tool to tool. Many tools return a vision reference frame. However, some tools return additional results items such as statistical information on how well the answer fits the actual data.

Vision Results Display

The Vision Results Window displays the "Results" properties for all of the tools evaluated in the previously executed Vision Process. In addition, for the position and orientation data, the results are displayed in both the vision coordinate system and the robot coordinate system utilizing the applicable camera calibration data. The information contained in this display is automatically updated each time a Vision Process completes execution.

A typical display that illustrates the results of a Vision Process that contains both a Finder and a Edge Locator Tool is shown in the following picture.

The following tables provide additional information on the window toolbar options and the data provided in each column of the results table.

Icon	Tool Tip Title	Description
🖫	Save	Stores the results data in a comma delimited file (*.CSV) that can be read into a spreadsheet application for further processing. A standard popup file window is displayed to allow the file name and disk directory path to be specified.
🗎	Copy to clipboard	Copies the contents of the results table to the system clipboard in a comma delimited (*.CSV) format. This information can then be pasted into another application.

Data Column	Description
Name	Specifies the name of the vision tool that generated the results.
Idx	For tools that return multiple sets of results, this is the index for the set of results that is displayed on a line. The index value goes from 1-n.
Vis X	For tools that return a position and orientation, this is the X & Y position in millimeters and the angle in degrees. These values are in the coordinate system of the camera's field-of-view.
Vis Y	
Vis Angle	
Rob X	For tools that return a position and orientation, this is the X & Y position in millimeters and the angle in degrees. These values are in the robot's world coordinate system and are the values that are transferred to a GPL program in a vision guidance application.
Rob Y	
Rob Angle	
Passed	Pass / Fail results of the tool Inspection. This is the same value as the **InspectPassed** property.
Insp Val	Actual property value that was tested by the tool Inspection. This is the same value as the **InspectActual** property.
Special Results	Each vision tool may have additional results that are unique to the tool type. This column provides a place for up to 20 special results per vision tool.
Errors and Warnings	Specifies if an error or warning condition occurred when the tool was executed. This is the same value as the **ResultErrorCode** property.

Application Status Bar

The bottom of the PreciseVision application window contains the Status Bar. This area continuously displays status information on the execution of the application. The Status Bar typically resembles the following.

The information contained in each of the columns of the Status Bar is described in the following table.

Col #	Data	Description
1	Server status	Indicates Ethernet traffic between PreciseVision and a Guidance Controller. Each time a message is received, the icon will flicker.
2	Image status	Indicates if the selected camera is in "live video" mode (bright light bulb) or a single still image has been captured.
3	Camera calibration file	Displays the name of the camera calibration file for the currently selected camera.
4	Camera calibration type	Displays the type of camera calibration that is in effect for the currently selected camera, e.g. Camera Only, Robot Vision, etc.
5	Process execution time	Indicates the total time required to execute the last Vision Process in seconds.
7	Camera coordinates and intensity reading	Shows the X/Y position of the mouse cursor in calibrated camera coordinates and the gray-level intensity value of the associated camera image pixel. This is useful for setting threshold and edge level values.
8	Project modified flag	Indicates if the currently loaded Vision Project has been modified in any way and must be saved to the disk in order to preserve the changes.

Preferences Panel

The Preferences Panel contains a number of application setup parameters that customize the operation and appearance of PreciseVision. Once saved, these preferences are automatically reloaded and put into effect each time the application is restarted.

The Preference Panels appears as shown below.
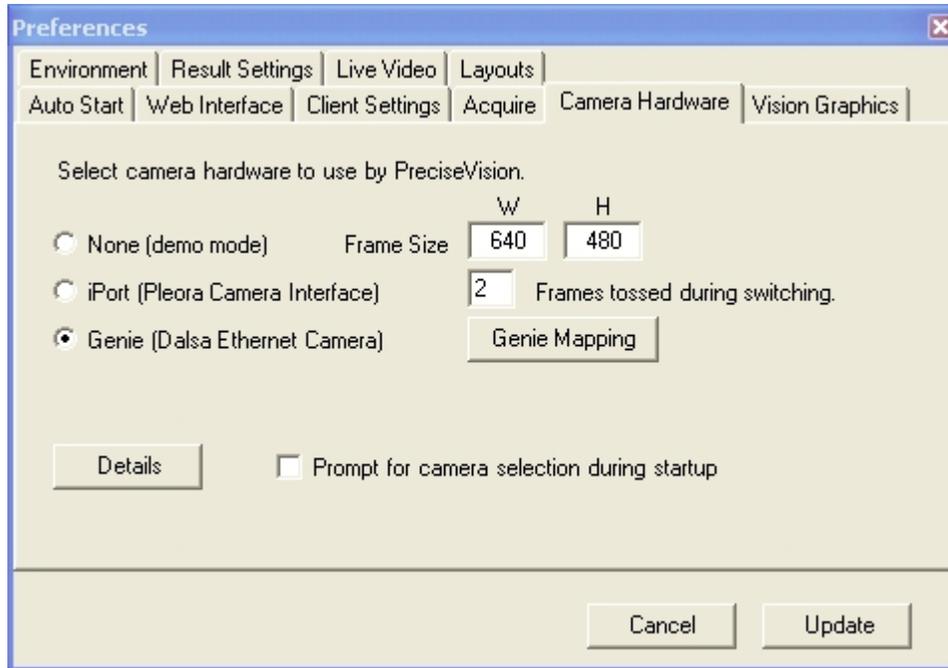
When any data in the Preferences has been modified, the **"Update"** button must be pressed before the change will take effect and be preserved.

In the following table, the configuration information presented on each tab of this panel is briefly described.

Tab	Description
Acquire	Defines the default disk file path for storing camera images when an Acquisition Tool is executed. This is a debugging convenience for logging images.
Auto Start	Specifies a Vision Project that will be automatically loaded each time PreciseVision is restarted. Once a Project is loaded, its Vision Processes are immediately available for execution by a GPL procedure executing on a Guidance Controller. This tab also contains the 'Startup Layout' specification. The specified layout will be automatically loaded each time PreciseVision is restarted. For production systems this would normally be set to the 'Run Mode' layout.
Camera Hardware	Specifies the type of camera hardware to be used to capture images. Normally, these parameters are defined during the software installation process. If the DALSA Genie camera is selected, the camera type and resolution is automatically detected and the frame size is automatically set.
Client Settings	Defines the Ethernet IP address for the Guidance Controller that is accessed during the camera calibration process. For normal operations, the controller IP address is automatically deduced from the commands that GPL sends to PreciseVision.
Environment	Controls whether the Main Toolbar and the Status Bar are displayed. Also permits adjustment of some diagnostic and demonstration parameters.

Live Video	Controls the rate at which the camera image is updated in Live Video mode. This allows the network traffic to be balanced when using PreciseVision. The update rate ranges from 1 to 30 frames per second (fps) with a default setting of 10. For cameras interfaced at 100Mb, a rate of 1 to 15 fps is typical. For cameras interfaced at 1000Mb, the refresh rate can be as high as 30 fps.
Result Settings	Enables the logging of Vision Tool Results to a disk log file for diagnostic purposes.
Vision Graphics	Enables and disables the display of Vision Tools and their Results in the Camera Display Window. Drawing these graphics has relatively little impact on the performance of the system, so the graphics are normally displayed.
Web Interface	Enables up to 3 web panels to be configured into the PV user interface as dockable controls. These panels can be utilized to integrate web pages of the Guidance Controller into the PV environment. Either the complete path for each web must be specified, e.g. "http://192.168.0.1/ROMDISK/web/Opr/master/masterfs.html" or just the path with the Client Guidance Controller without the Client's IP address, e.g. 'ROMDISK/web/Opr/master/masterfs.html'. In the latter case, the IP address of the Client Guidance Controller will be automatically assumed when displaying the panel.
Layouts	There are 3 predefined user-interface layouts of the dockable panels. These layouts provide an easy way to quickly switch between several display formats that are appropriate for different operating needs: production systems, editing sessions and calibration processes. If the 'Layout Filename' is left blank, PreciseVision will automatically use a predefined default layout stored in the 'Layout' folder of the installation directory. If the field contains a valid layout file, it will override these defaults. Click the 'Current' button in each selection to record the size of the window (width and height) automatically.

Genie Camera Mapping

On the "Camera Hardware" tab, a "Genie Mapping" button is active if the system is interfaced to a DALSA Genie camera. When this button is pressed, the following pop-up is displayed.
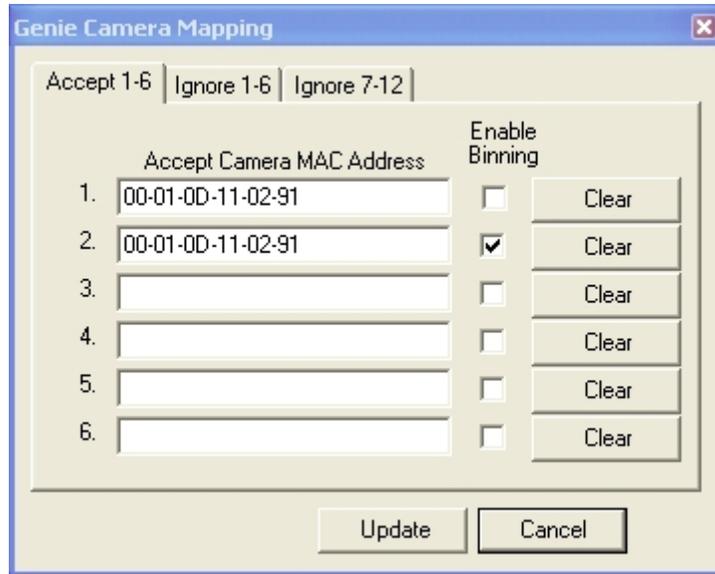
PreciseVision

The first time the DALSA Genie camera option is selected, PreciseVision will automatically scan for all available Genie cameras. When a camera is located, the camera list will be created. Each located camera is automatically assigned to a slot and will be identified by its MAC ID. The MAC ID is a unique number that is permanently assigned to every Ethernet compliant hardware component in the world.

Once the Genie camera list is created, a user can manually copy and paste MAC ID's to change the camera order. Each time PreciseVision is started, the physical cameras will be assigned to the same logical camera numbers based upon the contents of this table.

When PreciseVision starts up, if a new camera is found on the network, the new camera will automatically be assigned to the first available slot (logical camera number).

If a camera is disconnected or powered down, the slot for this camera is NOT cleared. This is important since there are unique calibration files for each physical camera. Also, all vision tools are linked to a specified logical camera number.

With the Genie camera, it is possible to assign the same camera to several camera slots. This allows different calibrations for the same camera if needed.

For a single camera configuration, simply start PreciseVision and the camera will be assigned to the first camera slot and can be referenced as logical camera #1.

Camera "Binning"

For DALSA cameras, the system supports "binning". At the camera hardware level, this process combines every other pixel horizontally and vertically and returns half of the standard number of pixels in each row and only every other row of pixels.

This process has the advantage of increasing the contrast in an image in the same manner as averaging two images. Also, since there are fewer pixels transmitted by the camera, the image acquisition time is reduced. However, the acquisition time is not reduced proportionally to the reduction in pixels. Although there are only 1/4th the number of pixels, for a 640x480 camera the speed improvement is only a few percent and for a 1392x1040 camera the speed improvement is about a factor of two. This is because the camera performs the vertical binning in the analog domain, which results in a 2x improvement, but the

54

horizontal binning is performed in the digital domain so all pixels in a row are still sampled. In addition, there is some overhead that further reduces the speed improvement. Nonetheless, for high resolution cameras, binning can be used to reduce the image capture time when lower resolution is acceptable.

To turn on binning for a logical camera, simply check the "Enable Binning" check mark next to the camera MAC address. To access a single camera in full resolution and binning mode, copy the MAC address to another logical camera number and enable binning on one of the logical cameras for the same physical camera. Whenever the logical camera with binning is accessed, binning will automatically be enabled.

When binning is turned on for a camera, the camera calibration for this logical camera must be re-computed. This is required since the pixel to mm scale factors will have changed.

Ignoring DALSA Cameras

For multiple cameras on the same Ethernet network that are being accessed by different PC's, there is an ignore list. Simply copy/paste any camera that you do NOT want PreciseVision to attach to during the startup into this list. This allows other PreciseVision applications on other PC's that are on the same network to use these cameras. The Ignore lists for up to 16 cameras are accessed by the tabs on the Genie Camera Mapping page.
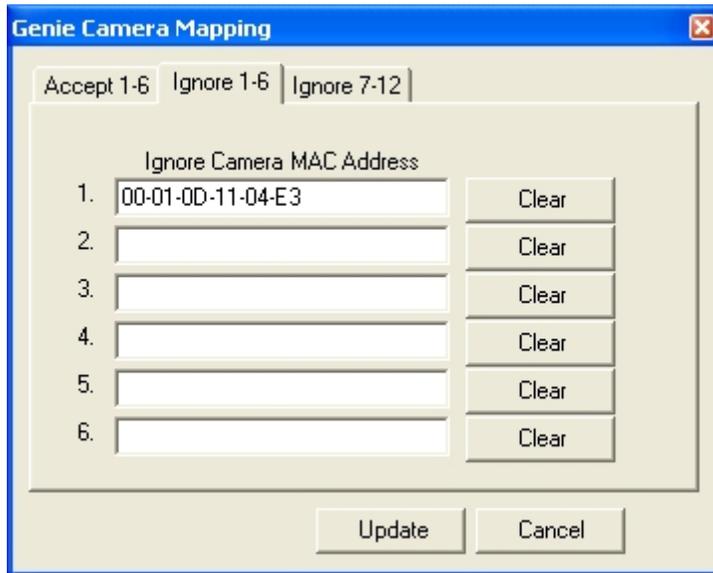
Vision Toolkit Description

Vision Toolkit Introduction

This chapter provides detailed information on each of the tools in the PreciseVision vision toolkit. These tools perform all of the runtime operations that are needed to acquire and process an image captured from an Ethernet camera.

Each instance of a tool is a vision object with a graphical front-end, a property list for configuring the tool for your desired operation, and a read-only list of results properties that can be used by other vision objects to create more complex machine vision and logical operations. The graphical front-ends allow the vision tools to be easily positioned over a captured image and greatly simplifies their training.

ToolKit Summary

The following table briefly summarizes the primary function of each of the vision tools. The "Type" broadly categories each tool as to whether it "Acquires" an image, "Finds" an object, "Measures" features, "Inspects" a region, "Computes" additional information, "Filters" image or object data or serves as a "Motion Sensor". Please see the detailed descriptions on the following pages for more ways in which these tools can be utilized.

Vision Tool	Type	Description
Acquisition Tool	Acquires	Captures an image from an Ethernet camera or loads it from a disk file, and stores the image into a frame buffer.
Arc Fitter Tool	Measures	Searches a specified region for arc edges and returns the arc that best fits the edges. Returns both the arc center point and radius.
Clear Grip Tool	Inspects	Verifies that there is no obstruction within the bounds of a defined window. Typically used to confirm that gripping a part will not result in a collision.
Color Pixel Window Tool	Inspects	Tests if the general color of a rectangular or circular region matches a trained color within a specified tolerance.
Connectivity Tool	Finds	Searches a region for binary blobs and returns the centroid and other features of each blob. In addition, this tool can return data that defines the perimeter of any located blob.
Edge Locator Tool	Measures	Detects edge points with sub-pixel accuracy along a linear path in a vision image and returns their positions. Alternately, this tool can return an array of the intensities along the linear path or a histogram of their values.

Finder Tool	Finds	Most powerful vision tool that identifies randomly placed parts in a camera image and returns their position, orientation and scaling to within sub-pixel accuracy.
Fixed Frame Tool	Computes	Places a reference frame at a fixed image coordinate or at a constant offset relative to another vision object. Can optionally index the frame in an X and/or Y grid pattern to repeatedly execute any linked tools.
Line Fitter Tool	Measures	Searches a specified region for edge points and returns the line that best fits the edges.
Line-Line Frame Tool	Computes	Determines the intersection of two lines defined by vision objects and returns a reference frame.
Pixel Window Tool	Inspects	Counts edges or binary pixels or collects gray-scale statistics within a rotated rectangular or circular region. Used for quickly detecting the presence of features or collecting general intensity information about a region.
Point-Line Frame Tool	Computes	Takes a point and a line from two vision objects and returns a reference frame.
Point-Point Line Tool	Computes	Computes a line given two points from vision objects.
Sensor Window Tool	Motion sensor	Detects motion in a window region. Ensures the scene is stable prior to other operations or waits until motion is detected. Can be used in place of physical sensors in flexible parts feeding where parts can take time to settle.
Tool Filter Tool	Filters	Takes the output from another tool that generates multiple sets of results, and returns a subset of the results based upon specified criteria.

Standard Tool Properties

To simplify their application, as much as practical, the vision tools make use of a common group of property names and associated operations. The following tables describe the standard tool properties that are common to many of the tools.

In these tables and the detailed tool descriptions, properties that are "read-only" have their name and data type dimmed. In addition, properties that have a counterpart property or method within the Guidance Programming Language include the GPL member name underlined and in italics bounded by square brackets preceded by "GPL:" and the class name, e.g. "*[GPL: Vision.ResultCount]*".

The **Identity Properties** specify the name of the tool and its basic tool type.

1. Standard Identity Properties			
Property Name	**Data Type**	**Range**	**Description**
Name	**String**	n/a	This is the unique name that must be assigned to each instance (copy) of a

PreciseVision

			general tool type. The name must follow the GPL variable naming conventions. That is, a name can be mixed case (upper and lower case characters) but when referenced by other tools, the names are not considered case sensitive (i.e. Abc and aBc refer to the same tool). Names must start with either a letter or an underscore "_" and can be followed by up to 127 additional letters, numbers and underscores. *In GPL, this is the name by which a tool is referenced.*
Type	String	n/a	This read-only property displays the basic type for the tool, e.g. VsLineFitter.

The **Placement / Size Properties** define the position, orientation and size of the tool in the coordinate system of the camera. In addition, if the tool is defined relative to another tool, the relative position and orientation can also be specified. Normally, it is easiest to graphically position and size each tool by dragging on the tool's handles displayed in the Camera Display Window. In the range specifications, "AOI" is an abbreviation for "Area of Interest" and refers to the size of the portion of the camera image that is being analyzed.

2. Standard Placement/Size Properties

Property Name	Data Type	Range	Description
Angle	Single	-360 to 360	These values specify the orientation (in degrees) and the position (in calibrated units, mm) of the center of the vision tool in the coordinates of the vision image. These values are automatically updated when the tool is graphically re-positioned and re-oriented during training and if the vision tool is placed relative to another vision tool during runtime.
XPos	Single	0 - AOI (mm)	
YPos	Single	0 - AOI (mm)	
Height	Single	0 - AOI (mm)	These values define the height and width of the vision tool in calibrated units (mm). These values are automatically updated when the tool is adjusted with the mouse or can be manually entered for more precise adjustments.
Width	Single	0 - AOI (mm)	
RelAngle	Single	-360 to 360	These values specify the orientation (in degrees) and the position (in calibrated units, mm) of the center of the vision tool relative to the vision object specified by **RelativeToolName**. If the vision tool is not relative to another vision object, these values are zero.
RelXPos	Single	0 - AOI (mm)	
RelYPos	Single	0 - AOI (mm)	

The **Operation Properties** define parameters that control the execution of the tool. For example, a tool might have an Operation Property that defines if it should screen out

58

black-to-white edges or the gray scale levels above which pixels are processed. Only the commonly used Operation Properties are placed in this group.

3. Standard Operation Properties			
Property Name	**Data Type**	**Range**	**Description**
Camera	**Integer**	1 - 6	Defines the number of the camera (and its associated frame buffer) on which the tool operates.
Relative-ToolName	**List**	n/a	If a tool is to be positioned relative to another tool, this is the name of the referenced tool. Tools can be relative to any vision tool that returns a vision coordinate position. By default tools are not relative to another tool and this property is blank.

The **Advanced Operation Properties** contain Operation Properties that are not normally modified and generally require a bit more experience to utilize. For these properties, their default settings will yield good tool performance for typical applications.

3A. Standard Advanced Operation Properties			
Property Name	**Data Type**	**Range**	**Description**

The **Inspection Settings Properties** permits a single scalar result of the vision tool to be automatically tested (inspected) to verify that it is within limits. A Pass / Fail property is set accordingly and a Pass / Fail label can be optionally displayed on the tool in the Camera Display Window.

4. Standard Inspection Settings Properties			
Property Name	**Data Type**	**Range**	**Description**
InspectType	**List**		Specifies the single scalar result that is to be inspected. The items in the list vary from one tool to the next. Select "None" to disable inspecting a result.
InspectLabel	**List**	NONE / PASS_AND_FAIL / PASS / FAIL	Indicates if a Pass / Fail label is to be displayed on the tool in the Camera Display Window based upon the results of the inspection. The items in the list indicate if no label should be displayed, only a Pass label, only a Fail label, or either a Pass or Fail label. The Pass indicator is displayed in green and the Fail is written in red. This selection is used for display purposes only and has no effect on the results of the inspection.

InspectMax	Single		Maximum and minimum values permitted for the item specified in **InspectType** to "Pass" the inspection. Values outside of this range are marked as "Failed".
InspectMin	Single		
InspectActual	Single		This is a read-only property that indicates the value of the result specified by **InspectType** that was tested during the inspection. *[GPL: VisResult.InspectActual]*
InspectPassed	List	Pass / Fail	Indicates if the inspection criteria was satisfied (Pass) or was not (Fail). Set to 'Pass' if no inspection is selected. If the inspection fails, any tool that references this tool, for example via the **RelativeToolName** property, will also have a failed InspectPassed although that tool will still be evaluated. *[GPL: VisResult.InspectPassed]*

The **Results Settings Properties** control the results that are computed by the vision tool. For example, if multiple sets of results can be computed by the tool, these properties can limit the number of sets of data computed and can select the set of data that is displayed in the **Results Properties**.

5. Standard Results Settings Properties			
Property Name	**Data Type**	Range	**Description**
MaxResults	**Integer**	-1 or 1 to n	For tools that provide multiple sets of results, this property can limit the maximum number of sets of results that are computed and returned. A negative 1 (-1) indicates that all possible results should be collected.
OffsetAngle	**Single**	degrees	These Offset values permit the results of a vision tool to be shifted and rotated from their standard location. This allows you to re-position the results of a tool to a position and orientation that is more meaningful to you. The orientation change is in degrees and the shift is in calibrated units (mm). See the **Fixed Frame Tool** for an alternate method for implementing this offset.
OffsetX	**Single**	mm	
OffsetY	**Single**	mm	
ResultSelect	**Integer**	1 - n	For tools that return more than one set of results, this property selects the set of results that is displayed in the **Results Properties**. This does not affect the actual results of the tool.
ShowResults	**List**	NONE / POINT /	Alters how the results of a tool is graphically displayed. Each vision tool

		LINE / FRAME	has a default method for display, e.g. a line or a frame. This property allows the graphical display to be changed. For example, a Fixed Frame Tool is normally displayed as a reference frame. However, if it is being used as a datum line or a point, its display can be changed to a line or a point for visual clarity. This property does not affect the actual results of the tool.

The **Results Properties** display a single set of results for the vision tool. In general, the results returned varies from tool to tool. Many tools return a vision reference frame. However, some tools return additional results items such as statistical information on how well the answer fits the actual data. If the tool computes more than one set of results and **MaxResults** is set to greater than 1, the displayed set of results will correspond to the setting of the **ResultSelect** property.

6. Results Properties			
Property Name	**Data Type**	**Range**	**Description**
ResultCount	**Integer**	0 - n	Number of sets of results the vision tool has returned. *[GPL: Vision.ResultCount]*
ResultErrorCode	**Integer**		Standard error code that indicates if the tool executed without an error. This is different from the **InspectPassed**, which indicates if the result was computed properly but its value was inside or outside of a specified limit. If **ResultErrorCode** is non-zero, any tool that references this tool, for example via the **RelativeToolName** property, will also fail with this same error code and will not be evaluated. *[GPL: VisResult.ErrorCode]*
ResultAngle	**Single**	-360 to 360	Orientation (in degrees) and position (in calibrated units, mm) of the center of the results computed by the vision tool in the coordinates of the vision image taking into consideration the value of the **Offset**'s in the **Results Settings Properties**. For robot guidance applications, these values are transformed by the vision-to-robot calibration conversion routines and then presented as *[GPL: VisResult.Loc.X, VisResult.Loc.Y and VisResult.Loc.Roll]*.
ResultXPos	**Single**	mm	
ResultYPos	**Single**	mm	

Acquisition Tool

Vision tool that captures an image from a camera or loads it from a disk file, and stores the image into a frame buffer.

Prerequisites

None

Properties

Property Name	Data Type	Range	Description
1. Identity			
Name	**String**	n/a	Standard Identity properties
Type	String	n/a	
3. Operation			
Camera	**Integer**	1 - 6	Standard Operation property
VideoGain	**Integer**	0.5 - 2.0	The **VideoGain** and **VideoOffset** scale and shift the grayscale value of each pixel as it is read in from the camera. This allows a narrow range of brightness values to be expanded to span the entire 256 grayscale value range.
VideoOffset	**Integer**	-127 - 127	
3A. Advanced Operation			
AcquireMode	**List**	NORMAL_ACQUIRE / ACQUIRE_AND_SAVE / PLAY_FROM_DISK / LIVE_VIDEO / SAVE_ONLY	"NORMAL_ACQUIRE" - acquires a single image from the specified camera and places it into a frame buffer. "ACQUIRE_AND_SAVE" - performs the same operation as "NORMAL_ACQUIRE" and then saves the image to a disk file. "PLAY_FROM_DISK" - restores an image from a disk file. The disk file is specified by the **AcquirePath** and **AcquirePrefix** (see below). If there are multiple numbered image files with the same prefix, each time the Vision Process is executed, the contents of the next sequential file will be copied to the image buffer. "LIVE_VIDEO" - continuously updates

			the vision display with the latest image from the camera. This is a system setup mode that facilitates setting the camera f-stop, focus, **VideoGain**, and **VideoOffset**. "SAVE_ONLY" - stores the contents of the current frame buffer to a disk file. This enables a GPL program to execute a vision process that saves selected frame buffers for later access.
AcquirePath	**String**	n/a	String that defines the path to the disk file when **AcquireMode** is set to "ACQUIRE_AND_SAVE" or "PLAY_FROM_DISK'". If blank (""), the path defined in the Preferences will be used. If the path does not contain a ':' (i.e. C:\...), **AcquirePath** will be appended to the path where the PreciseVision application is stored, e.g. "C:\Program Files\Precise Automation\PreciseVision #.#\"
AcquirePrefix	**String**	n/a	String that defines the disk file name (excluding the required .BMP extension) when **AcquireMode** is set to "'ACQUIRE_AND_SAVE" or "PLAY_FROM_DISK". If blank (""), the file name defined in the Preferences will be used. This string is combined with AcquirePath, a numerical index, and the ".BMP" file extension to generate the disk file name for saving or loading vision images.
ActiveLayer	**List**	Monochrome / RedLayer / GreenLayer / BlueLayer	Specifies the type of data to be loaded into the image frame buffer for processing by subsequent vision tools. For grayscale cameras, only "Monochrome" image buffers are available and each pixel in the frame buffer has a value from 0 to 255 that defines its grayscale intensity. For color cameras, the frame buffer can be loaded with one of four types of data: monochrome, red, green or blue. The monochrome mode produces the same results as a grayscale camera. The red, green and blue modes load the intensity data for a single color. For example, if "RedLayer" is selected, the image frame buffer will contain values that

			indicate the intensity (response) of only the red receptors of the camera. Tools that are applied to this image will only be operating on the red color layer of the camera. The one exception is the "Color Pixel Window" Tool. When this tool is executed, it automatically accesses all three of the RGB color layers of an image and is unaffected by the setting of the **ActiveLayer** property.
ExposureTime	**Single**		Time, in milliseconds, that the camera will integrate light on its imaging sensor. The longer the time, the more light is collected and the brighter the image. However, this will also lengthen the time required for the image acquisition. This parameter may not be supported on all hardware configurations. For the Pleora hardware configuration, this property is ignored.
GainBlue	**Single**	0-12 (set to -1 if not a color camera)	These properties scale the intensity values for each of the three color layers as they are read from the camera. This permits the color intensities to be balanced.
GainGreen	**Single**		
GainRed	**Single**		
ImageFile	**String**	n/a	Read-only string that displays the currently selected vision image file. This value is only displayed when the **AcquireMode** is set to "PLAY_FROM_DISK".
LoadFirstImage	**Boolean**	True / False	This is a convenience feature for setting the display of a sequence of images back to the first image and is only valid during the "PLAY_FROM_DISK" mode. That is, when this property is set to TRUE, the disk file that satisfies **AcquirePath** and **AcquirePrefix** and has the lowest index values will be loaded into the image buffer.

Remarks

This tool performs the basic image capture operation from an Ethernet color or monochrome camera or an analog monochrome camera interfaced to an Ethernet converter box, and stores an image plane in a frame buffer. Consequently, this tool is normally the first tool in each Vision Process.

Multiple cameras can be accessed by this tool and their gains and offsets can be controlled to optimize the brightness range for the field of view. To facilitate setting up a

camera's gain, offset, focus, and f-stop, the camera image can also be continuously acquired and displayed in the Camera Display Window.

As both a demonstration feature and as an aid in remotely diagnosing problems, this tool can be used to easily store captured images to a disk file and to reload images stored to the disk. To store images, the **AcquireMode** must be set to "ACQUIRE_AND_SAVE" or "SAVE_ONLY". To load files, the mode must be set to "PLAY_FROM_DISK". For any of these modes, the disk file name is constructed by combining the **AcquirePath** with the **AcquirePrefix** and an optional numerical index followed by the .BMP file extension. Each time that an **Acquisition Too**l is executed with "ACQUIRE_AND_SAVE" or "SAVE_ONLY" set, the system automatically increments the numerical index to create a new disk file. Likewise, each time that the **Acquisition** is executed with "PLAY_FROM_DISK", the system automatically searches the file folder for the file with the next larger numerical index. This automatic indexing allows a series of images to be conveniently generated or replayed.

Special Feature Buttons (located above the property editor)

Adjust Gains

Pressing this button displays a pop-up window that permits the property values for the monochrome image (**VideoGain, VideoOffset, ExposureTime**) and the color image (**GainRed, GainGreen, GainBlue**) to be manually adjusted while dynamically viewing the effect of the new settings. In addition, histograms of the various values can be simultaneously displayed.
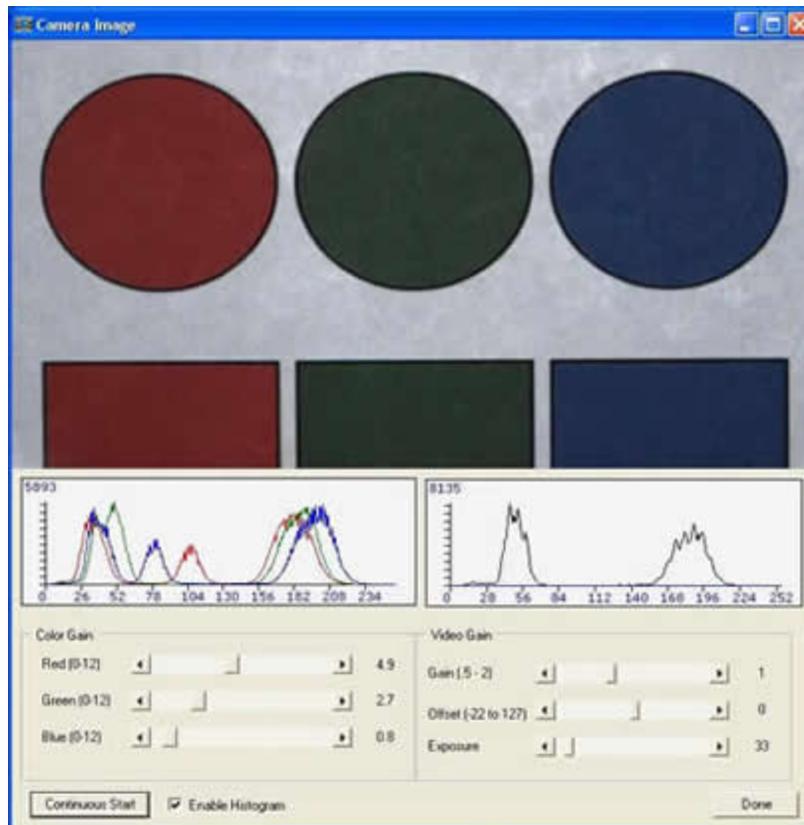
Select Acquire Path

Pressing this button displays a pop-up window that permits you to browse the PC's file structure to select the **AcquirePath** that specifies where image disk files are to be stored and retrieved.

Examples

If the **AcquireMode** is set to "PLAY_FROM_DISK", and the **AcquirePath** is set to "ExampleProjects", and the **AcquirePrefix** is set to "part2", and the folder that contains the PreciseVision application contains a folder "ExampleProjects" that has the following files:

part2_0001.bmp, part2_0002.bmp, part2_0003.bmp

Then, each time that the **Acquisition Tool** is executed, it will load the next successive file in the list above and will restart when it encounters the end of the list.

See Also

Vision Toolkit | Sensor Window Tool

Arc Fitter Tool

Vision tool that searches a specified region for arc edge points and returns the circle or arc that best fits the edges.

Prerequisites

None

Properties

Property Name	Data Type	Range	Description
1. Identity			
Name	**String**	n/a	Standard Identity properties
Type	String	n/a	
2. Placement/Size			
Angle	**Single**	-360 to 360	Standard Placement / Size properties. The X and Y values define the center of the tool and the center of the search region.
XPos	**Single**	0 - AOI (mm)	
YPos	**Single**	0 - AOI (mm)	
RelAngle	**Single**	-360 to 360	
RelXPos	**Single**	0 - AOI (mm)	
RelYPos	**Single**	0 - AOI (mm)	
Radius	**Single**	mm	Specifies the nominal radius of the arc or circle to be located. This defines the mid-range value for the radius search.
3. Operation			
Camera	**Integer**	1 - 6	Standard Operation properties
Relative-ToolName	**List**	n/a	
DarkOutside	**Boolean**	True / False	Defines whether the pixels outside of the arc or circle are darker than the interior pixels. For example, to locate a circle with light interior pixels surrounded by

			darker pixels, this parameter must be set to True.
EdgeThreshold	**Integer**	0 - 254	Specifies the threshold below which weak (low contrast) gradient edges are ignored. The lower this property is set, the more sensitive the system is in locating edges. Please see the description of the Edge Locator for more information on edge gradients. The default value is 30.
FindWhat	**List**	FITARC_FIND_CENTER / FITARC_FIND_RADIUS / FITARC_FIND_BOTH	Defines if the radius should be fixed and only the center of the arc varied to find the best match or if the center is to be fixed and the radius varied or if both should be varied.
AngularSize	**Integer**	0-360	Specifies the angular size in degrees of the arc to be located. A value of 360 indicates that a complete circle is to be located. Values less than 360 specify that a partial circle (arc) is to be located.
SearchRadiusRange	**Single**	mm	This property defines the limits of the center point and radius search regions. The search region is centered about the **XPos**, **YPos** and **Radius** values. So, if the **Radius** property is set to a value of N and the **SearchRadiusRange** is set to M, the radius can vary from N-M/2 to N+M/2 and the center point can vary by M in X and Y.
UseWhichEdge	**List**	FITARC_MIDDLE_EDGES / FITARC_INNER_EDGES / FITARC_OUTER_EDGES / FITARC_MAX_GRADIENTS	If multiple arcs exist within the search region, this parameter specifies which arc should be returned. The selected arc can be the one that is closest to the nominal radius, the arc closest to the minimum or maximum search radius or the arc that has the greatest contrast (maximum gradients).

3A. Advanced Operation

BiasFilter	List	..._BIAS_NONE / ..._BIAS_TO_LIGHT / ..._BIAS_TO_DARK	For arcs with an irregular or jagged edge, this parameter can be set to provide extra weight (importance) to edges that are closest to the light or dark side of the search region. For example, for an arc with a saw toothed pattern (e.g. a gear), this property could be set to fit the arc to the tips of the high or low points rather than along the center arc.
MaxEdgePoints	Integer	3 - n	Maximum number of edge point searches to perform. Higher numbers produce greater accuracy at the expense of execution speed. The minimum value of this parameter is 3 and this value is automatically limited to the number of pixels across the width of the tool. The default value is 30.
MaxIterations	Integer	0 - n	Maximum number of iterations for filtering. The filtering algorithm repeatedly removes outlier points, refitting the arc each time, until no more points need to be removed or the maximum number of iterations have occurred. A value of zero disables filtering. The default value is 5.
MinFilterDistance	Single	0 - n.nn	Absolute minimum filter distance in pixels. No edge points closer to the fitted arc than this are discarded during the iterative process. A minimum distance is needed because the standard deviation of distances to the fitted arc can be less than a pixel with a good image of a clean edge. The default value is 1.
SigmaFilter	Single	0 - n.nn	Filter width in units of standard deviations. This value is multiplied by the standard deviation of the

			edge points' distances to the fitted arc to compute the distance threshold beyond which edge points are removed during the iterative fitting process. The default value is 1.5.
4. Inspection Settings			
InspectActual	Single		Standard Inspection Settings properties
InspectLabel	**List**	NONE / PASS_AND_FAIL / PASS / FAIL	
InspectType	**List**	NONE / ..._RMS / ..._RADIUS / ..._NUM_EDGES_FOUND / ...NUM_EDGES_USED	
InspectMax	**Single**		
InspectMin	**Single**		
InspectPassed	List	Pass / Fail	
5. Results Settings			
OffsetAngle	**Single**	degrees	Standard Results Settings properties
OffsetX	**Single**	mm	
OffsetY	**Single**	mm	
ShowResults	**List**	NONE / FRAME / LINE / POINT	
6. Results			
ResultErrorCode	Integer		Standard Results properties
ResultAngle	Single	-360 to 360	
ResultXPos	Single	mm	
ResultYPos	Single	mm	
ResultNum-EdgesFound	**Integer**		Number of edge points found. *[GPL: VisResult.Info(1)]*
ResultNum-EdgesUsed	**Integer**		Number of edge points remaining after the filtering process. *[GPL: VisResult.Info(2)]*
ResultRadiusFound	**Single**		Radius of the located arc in mm. *[GPL: VisResult.Info(4)]*
ResultRMS	**Single**		Root mean square of the arc fit. *[GPL: VisResult.Info(0)]*

Remarks

The Arc Fitter searches the region defined by its green and blue outlined arc for edge points and fits an arc to the edge points using a least squares technique. By computing

70

the edge positions to a sub-pixel accuracy and by employing multiple such edges in its computation, this tool is able to generate very accurate sub-pixel results.

This tool is very useful for accurately locating holes or curved edges of objects or fiducials. Once found, these fit arcs can be used to compute center points between important features, for generating reference frames that can accurately locate an object or features in an object or for measuring the diameters of holes.

In order to make this tool more discriminating, the Arc Fitter only utilizes edges whose dark and light sides have roughly the same orientation as the dark (blue) and light (green) sides of the search area. So, when positioning this tool, you should place the center (nominal) arc of the search region at approximately the position and orientation that you expect to find the arc with the blue side on the anticipated dark side of the object. To aid in this operation, there is an 'Auto Size / Position' feature button above the property editor. This button will analyze the coarse position of the tool and snap it to the nearest arc/circle. The radius search range is automatically adjusted as well.

The operation of this tool is basically performed in two steps. First, all of the edges are extracted in the search region. Secondly, an arc is fit to the edges. Of the two operations, the tool spends most of its time extracting the edges. So, the speed and the accuracy of the Arc Fitter can be traded-off by adjusting **MaxEdgePoints**.   This property determines how densely the tool scans to detect edges. This parameter must be set to at least 3 in order to detect the minimum number of edges to define an arc. At most, one edge search is performed for each pixel along the width of the Fitter. Execution time increases roughly in proportion to the value of **MaxEdgePoints**. On the other hand, the accuracy of the tools increases approximately as the square root of **MaxEdgePoints**. That is, if you increase **MaxEdgePoints** by a factor of 4, the tool execution time will quadruple but the accuracy will only double.

Once the edges are found, the Arc Fitter will optionally perform an iterative fitting process to increase the robustness of the results. During each pass, edge points that are too distant from the arc (outliers) are discarded. This filters out edges that are not a part of the arc and whose inclusion would incorrectly offset the result. **MaxIterations** specifies the maximum number of fits to perform. After each arc fit, edges that are beyond **SigmaFilter** standard deviations are rejected, unless the edge is within **MinFilterDistance** pixels of the fit arc. The iterative process stops if (1) **MaxIterations** are performed, (2) no further edges are discarded, or (3) only two edge points remain.

At the conclusion of this tool, the center position and radius of the best fit arc is returned along with statistical data on the number of edges finally used and the root mean square of the arc fit, which indicates the quality of the final fit. This statistical data can be tested via the "Inspect Settings" to yield a pass/fail indication for the operation.

Special Feature Buttons (located above the property editor)

Adjust Edge Threshold

> Pressing this button displays a pop-up window that permits the value of the **Edge Threshold** property to be manually adjusted while dynamically viewing the effect of the new setting in the Camera Display. The display indicates the edges that satisfy the threshold criterion in white.

Auto Size / Position

Provides an easy way to position the tool. This button will compute the currently found arc/circle and automatically snap the tool to the exact coordinates of the found arc/circle. In addition the radius search range is adjusted proportionally to the arc/circle size.

Examples

In the following example, an Arc Fitter is utilized to accurately determine the position of a hole within an object. The best fit radius of the hole is computed and the resultant is compared to the acceptable range of values to generate a pass/fail inspection result.
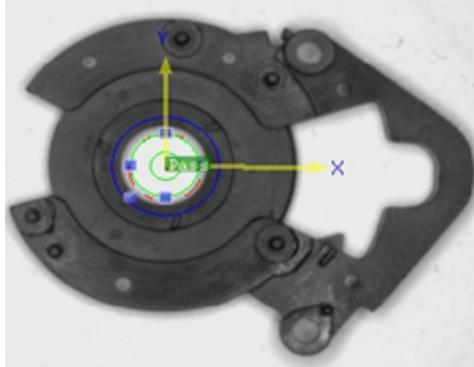
See Also

Vision Toolkit | Edge Locator | Line Fitter

Clear Grip Tool

Vision tool that verifies that there is no obstruction within the bounds of a defined window. This tool is typically used to confirm that gripping a part will not result in a collision.

Prerequisites

> None

Properties

Property Name	Data Type	Range	Description
1. Identity			
Name	**String**	n/a	Standard Identity properties
Type	String	n/a	
2. Placement/Size			
Angle	**Single**	-360 to 360	Standard Placement / Size properties. The X and Y values define the center of the verification window. The Height and Width define the dimensions of the window.
Height	**Single**	0 - AOI (mm)	
Width	**Single**	0 - AOI (mm)	
XPos	**Single**	0 - AOI (mm)	
YPos	**Single**	0 - AOI (mm)	
RelAngle	**Single**	-360 to 360	
RelXPos	**Single**	0 - AOI (mm)	
RelYPos	**Single**	0 - AOI (mm)	
3. Operation			
Camera	**Integer**	1 - 6	Standard Operation properties
Relative-ToolName	**List**	n/a	
EdgeThreshold	**Integer**	0 - 254	Specifies the threshold below which weak (low contrast) gradients edges are ignored. The smaller the number, the more sensitive the system is in locating edges. Please see the description of the Edge Locator for more information on edge gradients. The default value is 30.
4. Inspection Settings			

InspectActual	Single		
InspectLabel	**List**	NONE / PASS_AND_FAIL / PASS / FAIL	
InspectType	**List**	NONE / EDGE_COUNT	[Standard Inspection Settings properties](#)
InspectMax	**Single**		
InspectMin	**Single**		
InspectPassed	List	Pass / Fail	
5. Results Settings			
ShowResults	**List**	NONE / FRAME / LINE / POINT	[Standard Results Settings properties](#)
6. Results			
ResultErrorCode	Integer		
ResultAngle	Single	-360 to 360	[Standard Results properties](#)
ResultXPos	Single	mm	
ResultYPos	Single	mm	
ResultEdgeCount	**Integer**		Number of edge pixels found. *[GPL: VisResult.Info(0)]*
ResultTotalArea	**Integer**		Total number of pixels in the tested rectangular region. *[GPL: VisResult.Info(1)]*

Remarks

This tool specifies a rectangular region of an image and tests it to ensure that it is free of obstacles. This tool is typically used to verify that the fingers of a robot's gripper can reach in and pick up a part without hitting another part. This visual tool verifies that the region is clear by extracting edges within the region and counting the number of edge pixels. If an excessive number of edge pixels exist, this indicates that a part or obstruction is in the region.

The Clear Grip region is normally defined relative to another vision tool, e.g. a Finder, that is used to locate the part. In fact, it is common for two or more clearance regions to be defined for a part, one for each of the gripper's fingers. When the Clear Grip tool is processed, all of the pixels within the region whose gray-scale gradient value exceeds the **Edge Threshold** are counted as edge pixels. By setting the **InspectType** to EdgeCount, if the number of edges falls outside of the **InspectMax** and **InspectMin** limits, this tool will signal a failure.

Special Feature Buttons (located above the property editor)

Adjust Edge Threshold

Pressing this button displays a pop-up window that permits the value of the **Edge Threshold** property to be manually adjusted while dynamically viewing the effect of the new setting in the Camera Display. The display indicates the edges that satisfy the threshold criterion in white.

Examples

In the following example, the Clear Grip tool is used to test if randomly oriented and positioned parts can be safely grasped. As each part is located by a Finder, two Clear Grips tools are tested, one on each side of the part. If the grip zone does not contain a portion of another part, the zone is marked as passed. However, in the sample picture, one zone in the center of the image is partially occluded by another part and that zone is marked as failed. This would prevent the robot from attempting to pick up the part in the center and potentially damaging the adjacent part or the gripper.
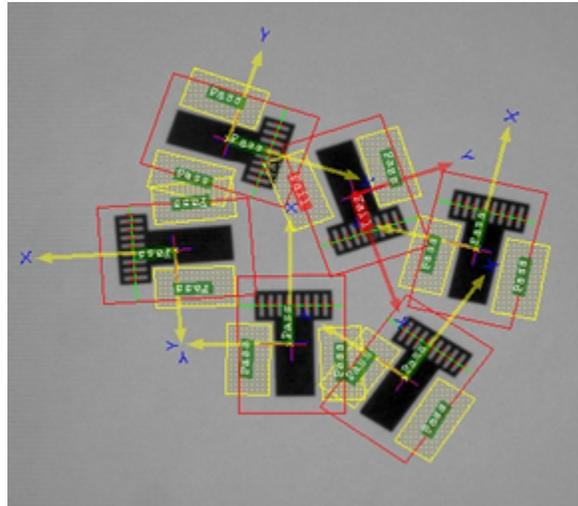
See Also

Vision Toolkit | Finder Tool

Color Pixel Window Tool

Vision tool that tests if the color of a region matches a trained color. Within each of the RGB layers of a color image, this tool computes the mean intensity value for a specified region and determines if each mean is within an acceptable tolerance of the trained value for the color layer.
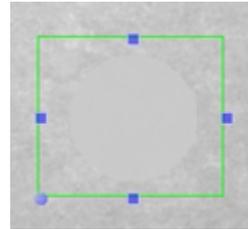
Prerequisites

Requires a color camera.

Properties

Property Name	Data Type	Range	Description
1. Identity			
Name	**String**	n/a	Standard Identity properties
Type	String	n/a	
2. Placement/Size			
Angle	**Single**	0 (*non rotating*)	Standard Placement / Size properties. The X and Y values define the center of the region to be processed. The Height and Width define the dimensions of the region.
Height	**Single**	0 - AOI (mm)	
Width	**Single**	0 - AOI (mm)	
XPos	**Single**	0 - AOI (mm)	
YPos	**Single**	0 - AOI (mm)	
RelAngle	**Single**	0	
RelXPos	**Single**	0 - AOI (mm)	
RelYPos	**Single**	0 - AOI (mm)	
3. Operation			
Camera	**Integer**	1 - 6	Standard Operation properties
Relative-ToolName	**List**	n/a	
ToleranceBlue	**Integer**	0-100%	Define the tolerances that must be satisfied between
ToleranceGreen	**Integer**	0-100%	

ToleranceRed	**Integer**	0-100%	the computed mean intensity values of the specified region in each color layer and the corresponding trained mean values. For example, if the **ToleranceBlue** is set to 5%, the computed mean of the pixel values in the specified region of the blue color layer must be within +/-5% of the **InspectNominalBlue** value for the inspection to return "Pass"..
ToolShapeType	**List**	RECTANGULAR / ROUND / SQUARE	Shape of the window in which the pixels are analyzed. A square window has its aspect ratio forced to 1:1.
4. Inspection Settings			
InspectActualBlue	Integer	0-255	Standard Inspection Settings properties. In the RESULT_FIXED mode, the tolerance comparison is performed with the raw nominal and actual mean intensity values. This can be used when the lighting is very controlled and the product color does not vary. In the RESULT_NORMALIZED mode, the nominal and actual mean values are adjusted each time the tool is executed to compensate for uniform variations in the color intensity values. This enables the color matching to be robust even when there is some variation in the lighting.
InspectActualGreen	Integer		
InspectActualRed	Integer		
InspectLabel	**List**	NONE / PASS_AND_FAIL / PASS / FAIL / VALUE_REAL / VALUE_INTEGER	
InspectType	List	RESULT_NORMALIZED / RESULT_FIXED	
InspectPassed	List	Pass / Fail	
InspectNominalBlue	Integer	0-255	These properties define the nominal mean values for the three color layers. These values are compared to the actual mean values for the color layers in the region of interest to determine if the color matches the expected value. These values are typically taught by pressing the Train Color button (see below).
InspectNominalGreen	Integer		
InspectNominalRed	Integer		
5. Results Settings			

OffsetAngle	Single	degrees	Standard Results Settings properties
OffsetX	Single	mm	
OffsetY	Single	mm	
ShowResults	List	NONE / FRAME / LINE / POINT	
6. Results			
ResultErrorCode	Integer		Standard Results properties
ResultAngle	Single	0	
ResultXPos	Single	mm	
ResultYPos	Single	mm	
ResultMeanBlue	**Single**	0-255	Mean color value of pixels in specified region. *[GPL: VisResult.Info(2)]*
ResultMeanGreen	**Single**	0-255	Mean color value of pixels in specified region. *[GPL: VisResult.Info(1)]*
ResultMeanRed	**Single**	0-255	Mean color value of pixels in specified region. *[GPL: VisResult.Info(0)]*
ResultPercentBlueMatch	**Integer**	0-100	Match percentage for a color layer *[GPL: VisResult.Info(5)]*
ResultPrecentGreenMatch	**Integer**	0-100	Match percentage for a color layer *[GPL: VisResult.Info(4)]*
ResultPrecentRedMatch	**Integer**	0-100	Match percentage for a color layer *[GPL: VisResult.Info(3)]*

Remarks

This tool compares the general color of a region of interest with a taught color and returns a Pass/Fail indication of whether the colors matched within a specified tolerance. The region can be defined as a rectangular, square or circular shape. The color of the region is determined by computing the mean intensity value for each of the three color layers (RGB). These actual mean values are compared against three taught mean values. If all three pairs of number are within specified % tolerances, the colors are considered to be a match and this tool returns a pass indication.

To improve the robustness of this tool when the lighting conditions may vary, the **InspectionType** can be set to RESULT_NORMALIZED. In this mode, the taught nominal values and the actual computed mean intensity values are adjusted for the over-all brightness of the region of interest.

This tool always accesses all three color layers of the captured color image independent of which layer the picture **Acquisition Tool** loaded into the current frame buffer.

To assist in teaching this tool, the center of the tool is always displayed with a semi-transparent circular area that is tinted with the mean color for the region of interest. This is illustrated in the following picture where the center is tinted in blue.
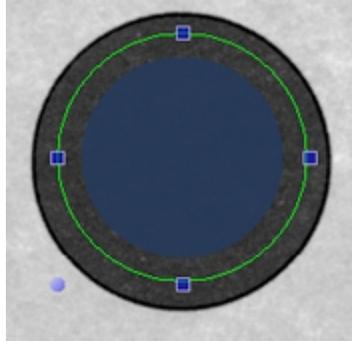
In addition, the camera display window can always be switched into color display mode by pressing on the "Toggle Color Display Mode" button in the main Toolbar.

Special Feature Buttons (located above the property editor)

Train Color

Pressing this button assists in teaching the color for the region of interest. After confirmation, the values for the **InspectNominalRed, InspectNominalGreen, InspectNominalBlue** properties are set to the current mean values of the selected region of the currently displayed image.

Examples

See Also

Vision Toolkit | Pixel Window

Connectivity (Blob) Tool

Vision tool that searches a specified region for binary blobs and returns the centroid and other features of each blob. In addition, this tool can return data that defines the perimeter of a located blob.

Prerequisites

None

Properties

Property Name	Data Type	Range	Description
1. Identity			
Name	**String**	n/a	Standard Identity properties
Type	String	n/a	
2. Placement/Size			
Angle	**Single**	0 (*non rotating*)	Standard Placement / Size properties. The X and Y values define the center of the tool and the center of the search region. The Height and Width specify the size of the search region.
Height	**Single**	0 - AOI (mm)	
Width	**Single**	0 - AOI (mm)	
XPos	**Single**	0 - AOI (mm)	
YPos	**Single**	0 - AOI (mm)	
RelAngle	**Single**	0	
RelXPos	**Single**	0 - AOI (mm)	
RelYPos	**Single**	0 - AOI (mm)	
3. Operation			
Camera	**Integer**	1 - 6	Standard Operation properties
Relative-ToolName	**List**	n/a	
MaxArea	**Integer**	0 - n	Specifies the maximum number of pixels that a valid blob can contain. Any blob that consists of a greater number of pixels will be ignored.
MinArea	**Integer**	0 - n	Specifies the minimum number of pixels that a valid blob must contain.

			Any blob that consists of a smaller number of pixels will be ignored.
ObjectColor	List	DARK_OBJECT / LIGHT_OBJECT	After the image area is converted to binary, defines whether connected groups of black or white pixels should be considered as blobs.
Threshold	Integer	0 - 254	Specifies the threshold below which gray-scale pixels will be set to black and above which pixels will be set to white. When this tool is created, the system will automatically set this property to a reasonable value for the full vision window. This value can also be updated using the special feature button described below.
3A. Advanced Operation			
MinHoleArea	Integer	0 - n	Minimum number of pixels required before a hole in a blob is retained and not filled in. If this property is set to 1 (default), all holes are included in blobs regardless of their size. If this value is set to the total area of the image, holes will always be deleted.
SampleRate	Integer	0 - n	When the tool is set to return PERIMETER information, this properly specifies how many pixels to skip along the perimeter between each returned "result" position. A value of '0' will not skip any pixels and every perimeter point is returned.
4. Inspection Settings			
InspectActual	Single		[Standard Inspection Settings properties]
InspectLabel	List	NONE / PASS_AND_FAIL / PASS / FAIL	

InspectType	List	NONE / ..._RMS / ..._ANGLE / ..._NUM_EDGES_FOUND / ..._NUM_EDGES_USED	
InspectMax	Single		
InspectMin	Single		
InspectPassed	List	Pass / Fail	

5. Results Settings			
MaxResults	Integer	-1 or 1 to n	Standard Results Settings properties.

MaxResults defines the number of blobs or perimeter points to find each time this tool is executed. By default, this property is set to -1, find all possible blobs or perimeter points.

See below for a description of **ResultMode**. |
OffsetAngle	Single	degrees	
OffsetX	Single	mm	
OffsetY	Single	mm	
ResultMode	List	BLOB_CENTER / MIN_RADIUS / MAX_RADIUS / MAJOR_AXIS / MINOR_AXIS / PERIMETER	
ShowResults	List	NONE / FRAME / LINE / POINT	

6. Results			
ResultCount	Integer	0 - n	Standard Results properties

ResultCount indicates the number of blobs located in non-PERIMETER mode. In PERIMETER mode, this is sum of all perimeter points found for all blobs. |
ResultErrorCode	Integer		
ResultAngle	Single	-360 to 360	
ResultXPos	Single	mm	
ResultYPos	Single	mm	
ResultBlobArea	Integer	pixels	Total area of the blob in pixels excluding any holes within the blob. *[GPL: VisResult.Info(3)]*
ResultNumPerBlobs	Integer	0 - n	For PERIMETER mode, this is the total number of blobs located and **ResultCount** indicates the sum of all perimeter points available. For non-PERIMETER mode, this value is undefined and **ResultCount** indicates the total number of blobs located. *[GPL: VisResult.Info(2)]*
ResultObjectIdx	Integer		Each blob in a region is assigned an index. In PERIMETER mode, each

				returned *VisResult* represents a single perimeter point. If multiple blobs are located, this property can be monitored to determine when the perimeter points pertain to the next blob. *[GPL: VisResult.Info(0)]*
ResultMajorAxisAngle	**Single**		-360 to 360	Angle *[GPL: VisResult.Info(8)]* and Length *[GPL: VisResult.Info(9)]* of the major axis of the best fit ellipse for the blob. This axis provides a means for characterizing the orientation of an arbitrary shape unambiguously within +-90 degrees. By definition, this ellipse has the same 2nd moments of inertia as the blob.
ResultMajorAxisLength	**Single**		mm	
ResultMinorAxisAngle	**Single**		-360 to 360	Angle *[GPL: VisResult.Info(10)]* and Length *[GPL: VisResult.Info(11)]* of the minor axis of the best fit ellipse for the blob. This axis provides a means for characterizing the orientation of an arbitrary shape unambiguously within +-90 degrees. By definition, this ellipse has the same 2nd moments of inertia as the blob.
ResultMinorAxisLength	**Single**		mm	
ResultMaxRadiusAngle	**Single**		-360 to 360	Angle *[GPL: VisResult.Info(6)]* and Length *[GPL: VisResult.Info(7)]* of the maximum radius for the blob. This radius is measured from the blob centroid to the point on the perimeter that is furthest from the centroid. This provides a means for characterizing the orientation of an arbitrary shape. However, the uniqueness of this radial line is a function of the shape.
ResultMaxRadiusLength	**Single**		mm	

ResultMinRadiusAngle	Single	-360 to 360	Angle *[GPL: VisResult.Info(4)]* and Length *[GPL: VisResult.Info(5)]* of the minimum radius for the blob. This radius is measured from the blob centroid to the point on the perimeter that is closest to the centroid. This provides a means for characterizing the orientation of an arbitrary shape. However, the uniqueness of this radial line is a function of the shape.
ResultMinRadiusLength	Single	mm	

Remarks

The Connectivity tool converts the region defined by its green outlined rectangle to a binary image and groups adjacent ("connected") pixels with the correct **ObjectColor** into blobs. Any blobs that do not satisfy the specified size limits as well as any interior blobs (i.e. blobs within blobs) are not returned. For all valid blobs, the position of their centroid and other key features are computed and returned as the results of this tool.

This tool is similar to the Finder Tool in that multiple objects can be identified and located. However, this is a less discriminating and less powerful tool than the Finder, but it is nonetheless very useful. In particular, Connectivity should be used for locating objects that do not have a fixed boundary and in applications that have high contrast with visually simple images. Parts that vary in size and shape are easily located using this tool. In addition, this tool has the ability to return the perimeter of any blob as a series of individual results points. This perimeter retrieval feature allows an application to easily trace an object, which is useful in the textile or automated cutting industries.

If the vision system must differentiate between objects that differ by subtle features, locate objects that touch other objects, locate objects that are not clearly differentiated from the background, or operate when thresholding may not be appropriate or robust, the Finder Tool should be utilized in place of Connectivity.

With regard to the performance of the Connectivity tool, even though this tool operates on binary images, the X & Y position of the centroid of blobs is normally accurate to within a fraction of a pixel. This is due to the fact that the centroid is computed by taking into consideration the positions of all of the perimeter pixels. In general, the larger the blob, the more accurate the centroid position. Consequently, the centroid position is a very good measure of where the blob is located.

In addition to the blob centroid, each blob has several other features computed as indicated in the following drawing.
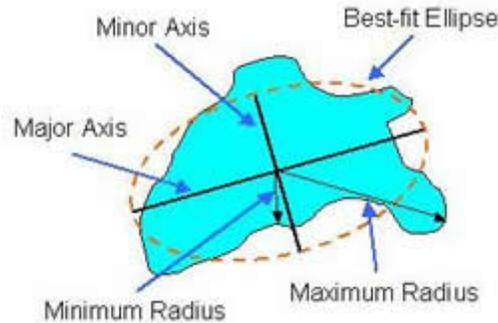
These additional features can be used to determine the orientation of the blob. Depending upon the exact shape of the object, some of these features will be more useful than others. For example, if the blob is a rectangle, all four corners will be at the same maximum radius from the centroid, so the **ResultMaxRadiusAngle** can return any one of four values, all of which are correct. In many cases, the orientation of the major axis of the best-fit ellipse (**ResultMajorAxisAngle**) can provide a very good orientation estimation. But, it should be kept in mind that this parameter always has a 180 degree ambiguity due to the symmetry of an ellipse.

The **ResultMode** property selects which of the properties is utilized to compute the orientation of each blob or if the orientation is to be fixed (BLOB_CENTER).

If the **ResultMode** property is set to PERIMETER, this tool returns a series of results, one for each desired point on the perimeter of each located blob. Each *VisResult* contains the X & Y position of a point on the perimeter with the orientation angle set to be perpendicular to the perimeter of the blob. In this mode, the **SampleRate** specifies if some pixels on the perimeter are to be skipped. For example, if the **SampleRate** is set to 3 and two blobs are located, the coordinates of every 3rd pixel on the perimeter of the first blob will be returned in the *VisResult* and **ResultObjectIdx** will be set to 1. After the last perimeter position of the first blob is transmitted, every 3rd pixel of the perimeter of the second blob will be returned with the **ResultObjectIdx** set to a value of 2.

Special Feature Buttons (located above the property editor)

Automatic Threshold

The Connectivity tool operates on a thresholded binary image. This requires a specific gray level to be entered in the **Threshold** property to define which pixels are to be converted to black (below the **Threshold**) and white (above the **Threshold**). To help set this value, there is an "Automatic Threshold" button available above the property editor. When this button is pressed, the **Threshold** property is automatically set to the system's best estimate for the tool's search region.

If the **Threshold** is properly set, a thin red boundary should will be drawn along the perimeter of each located object. If required, manually adjust the **Threshold** value to ensure the displayed red boundary is properly aligned with the object boundary.

Adjust Threshold

Pressing this button displays a pop-up window that permits the value of the **Threshold** property to be manually adjusted while dynamically viewing the effect of the new setting in the Camera Display. The displayed image shows all pixels converted to black or white, where the white pixels are the ones that are above the threshold.

Examples

In the following example, a single part is located using a Connectivity (blob) tool. The maximum radius is used to determine the orientation of the part (**ResultMode** set to MAX_RADIUS). The red outline illustrates the perimeter of the object as computed by the tool. The intersection of the X and Y axes denotes the computed centroid of the blob. The X-axis is aligned with the radial line to the perimeter point that is furthest away from the centroid. For this part, the maximum radius point yields a unique orientation angle for the object.
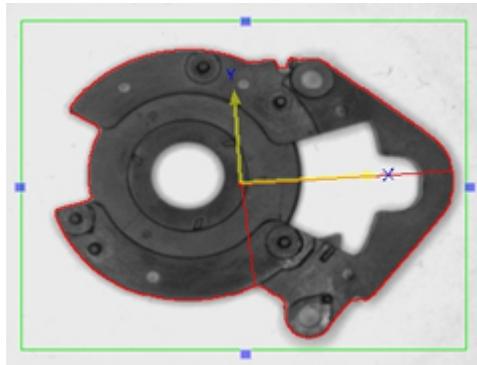
In the following example, the **ResultMode** is set to PERIMETER and the Connectivity tool returns the locations of the points along the perimeter of the object. In this example, the **SampleRate** has been set to 20, so a yellow "X" is drawn every 20 pixels along the perimeter representing each of the returned values.
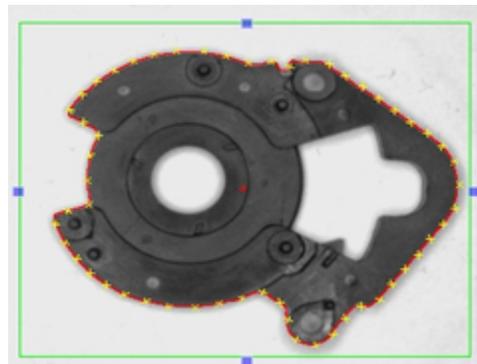
In this following example, a Fixed Frame tool has been added to the vision process. The Fixed Frame has its **RelativeToolName** set to the name of the Connectivity tool. The Connectivity tool is once again set into PERIMETER mode. Since the orientation of each of the perimeter results is perpendicular to the perimeter of the blob, the Fixed Frame can be used to create a series of points that are a fixed distance inside of the boundary of the blob. In the picture below, the display of the blob perimeter points has been turned off and the position of each Fixed Frame is being displayed as a yellow "X".
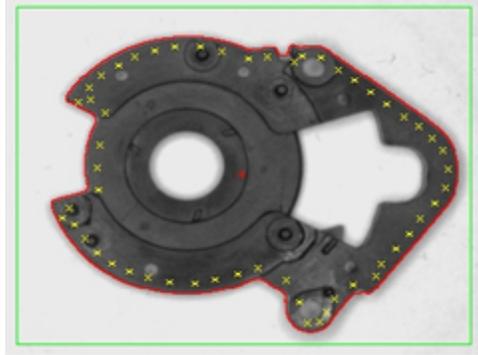
See Also

Vision Toolkit | Finder

Edge Locator Tool

Vision tool that detects edge points with sub-pixel accuracy along a linear path in a vision image and returns their positions. Alternately, this tool can return an array of the intensities along the linear path or a histogram of their values.

Prerequisites

None

Properties

Property Name	Data Type	Range	Description
1. Identity			
Name	**String**	n/a	[Standard Identity properties](#)
Type	String	n/a	
2. Placement/Size			
Angle	**Single**	-360 to 360	[Standard Placement / Size properties.](#) The X and Y values define the center of the tool.
XPos	**Single**	0 - AOI (mm)	
YPos	**Single**	0 - AOI (mm)	
RelAngle	**Single**	-360 to 360	
RelXPos	**Single**	0 - AOI (mm)	
RelYPos	**Single**	0 - AOI (mm)	
Length	**Single**	0 - AOI (mm)	Defines the length of the linear path that is tested for edges.
3. Operation			
Camera	**Integer**	1 - 6	[Standard Operation properties](#)
Relative-ToolName	**List**	n/a	
EdgeThreshold	**Integer**	0 - 254	Specifies the gray scale threshold value used to detect edges or (in the case of gradient values) the threshold for the rate of change of the gray scale values. The smaller the number, the more sensitive the system is in locating edges. The default value is 30.
3A. Advanced Operation			

DesiredAlg	**List**	EDGE_BINARY / EDGE_GRAYTHRES / EDGE_SUBPIX / EDGE_HIGH_-PRECISION_SUBPIX	EDGE_BINARY: identifies edge transitions in a binary image. EDGE_GRAYTHRESH: identifies an edge position whenever the gray scale value crosses a specified threshold value. EDGE_SUBPIX: computes the positions of edges to a sub-pixel accuracy based upon the rate of change (gradient) of the gray scale values. This is the default setting. EDGE_HIGH_PRECISION_SUBPIX: similar to the sub-pixel mode except that a more precise interpolation technique is employed that provides more accurate sub-pixel edge locations at the cost of a small increase in execution time. In rare cases where very fine features (e.g. 2 or 3 pixel bumps) are contained in the edge and need to accurately accounted for, EDGE_SUBPIX may produce more accurate results since that method uses a smaller pixel region to compute its results.
DesiredGrads	**List**	EDGE_ALLGRADS / EDGE_POSGRADS / EDGE_NEGGRADS	This parameter specifies whether all edges are to be detected (EDGE_ALLGRADS) or only edges that occur when the gray scale value goes from dark-to-light (EDGE_POSGRADS) or only edges when the gray scale value goes from light-to-dark (EDGE_NEGGRADS). Detecting only negative or positive edges helps disambiguate the edge being detected or can eliminate double edges that occur when thin dark or light features are encountered. Processing starts at the square resizing handle and ends at the round rotation handle.
MaxEdgePoints	**Integer**	1 - n	Maximum number of edge points to return.
4. Inspection Settings			
InspectActual	Single		[Standard Inspection Settings properties.](#)
InspectLabel	**List**	NONE / PASS_AND_FAIL / PASS / FAIL	All of the inspected results should be self explanatory except perhaps the

InspectType	**List**	NONE / NUMBER_OF_POINTS / DISTANCE_TO_-FIRST_POINT / DISTANCE_TO_-LAST_POINT / DISTANCE_TO_-MID_POINT / DISTANCE_FROM_-FIRST_TO_LAST / GRAY_AVERAGE / GRAY_MIN / GRAY_MAX	"GRAY_...", which test the average or min/max grayscale values along the length of the entire tool.
InspectMax	**Single**		
InspectMin	**Single**		
InspectPassed	**List**	Pass / Fail	
5. Results Settings			
MaxResults	**Integer**	-1 or 1 to n	
OffsetAngle	**Single**	degrees	
OffsetX	**Single**	mm	
OffsetY	**Single**	mm	[Standard Results Settings properties](#)
ResultSelect	**Integer**	1 - n	
ShowResults	**List**	NONE / FRAME / LINE / POINT	
ResultMode	**List**	RESULT_EDGE_POINTS / RESULT_FIRST_POINT / RESULT_LAST_POINT / RESULT_MID_POINT / RESULT_PASS_-THRU_SOURCE / RESULT_HISTOGRAM / RESULT_GRAYLEVEL	Defines whether all edge points or selected edge points are to be returned in the results. The "RESULT_PASS_THRU_SOURCE" is a special mode where this tool returns the "Results" of its parent tool pointed to by **RelativeToolName**. This permits a process to add one or more Edge Locators after it has computed the position of a part and use the Locators to perform additional visual verification tests, but still have the process return the "Results" location of the parent tool. Please see the "PassFailEdgeCount" demonstration program that was shipped with PreciseVision for an example of the use of this feature. The "RESULT_GRAYLEVEL" and "RESULT_HISTOGRAM" are special modes that return an array containing either the intensity (grayscale) value for each pixel along the length of the tool or the computed histogram of the distribution of intensity values. If either of these modes are selected, edge points are not detected.

6. Results			
ResultCount	Integer	0 - n	
ResultErrorCode	Integer		
ResultAngle	Single	-360 to 360	[Standard Results properties](#)
ResultXPos	Single	mm	
ResultYPos	Single	mm	

Remarks

This tool finds object boundaries, i.e. edges, along a linear path in an image. The edge positions can be used to measure the dimensions of objects or detect if key features are present or absent. The edges can be detected in a number of different modes (e.g. threshold, sub-pixel, etc). Alternately, the raw pixel values or a histogram of the grayscale values of the pixels along the path can be returned. In other vision systems, this type of tool is sometimes called a "caliper" or "ruler".

Locating Edge Points

When locating edge points, this tool operates by analyzing pixels along the specified line. Processing starts at the square resizing handle and ends at the round rotation handle. In cases where you wish to compute the position of a single straight or smoothly curved edge, the Line Fitter and Arc Fitter tools will yield more accurate results because they compute an edge position utilizing many more pixel values.

The primary mode of operation for this tool is specified by the **Desired Alg** property. If this parameter is set to "EDGE_GRAYTHRESH", an edge is detected whenever the gray scale value crosses the threshold specified by **EdgeThreshold**. The graph below shows a typical example of how the gray scale values might appear along the length of an Edge Locator. In this case, if the threshold is set to 100, two edges will be detected (Crossing #1 and #2) where the gray scale values cross the specified threshold value.
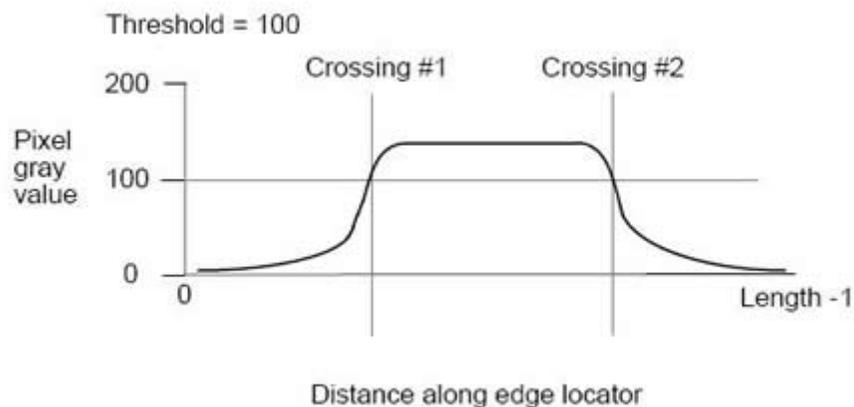
If the **Desired Alg** property is set to "EDGE_SUBPIX", edge positions are computed to a sub-pixel accuracy by first computing the rate of change ("gradients") of the gray scale values along the length of the Edge Locator. For each region where the gradients

exceed the **Edge Threshold** value, the peak of the gradient is computed with sub-pixel accuracy and returned as the location of the edge.

Threshold = 50
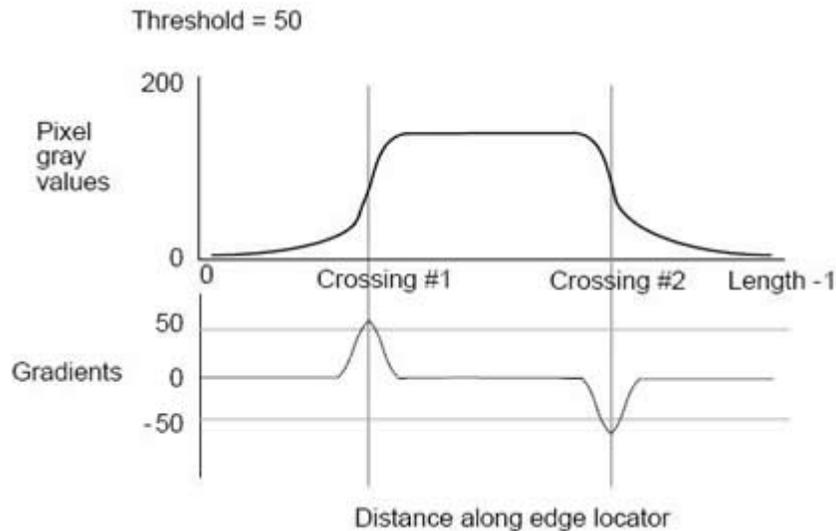
If **Desired Alg** is set to "EDGE_HIGH_PRECISION_SUBPIX", a higher-resolution sub-pixel position is computed than the ""EDGE_SUBPIX" method by employing an even more accurate (and costly) sub-pixel determination method. This method analyzes a longer gradient neighborhood along the path of the Edge Locator.

Finally, if **Desired Alg** is set to "EDGE_BINARY", 0-to-1 or 1-to-0 transitions are detected in a binary image.

After all of the edges have been determined, this tool can return all of the located edges or a selected edge based upon the setting of the **ResultMode** property.   Also, the standard "Inspection Settings" can be used to yield Pass / Fail results based upon the number of edges detected, various distance relationships or the average grayscale value along the entire length of the tool.

<u>**Computing Intensity Histogram**</u>

If the **ResultMode** is set to "RESULT_HISTOGRAM", rather than computing edge points, this tool counts the number of pixels along the specified line that has each of the possible 256 grayscale values and returns the pixel counts. This count of the grayscale distribution of pixels is called an intensity "Histogram".   It is useful for determining the prevalent grayscale values and the intensity differences between the peak values.

If you are executing GPL version 2.0 or later, an array of 256 counts (0[black] - 255[white]) can be fetched using the **VisResults Info** property. This array of results is also displayed in the Vision Results panel of the PV user interface in the "Special Results" section and can be recorded if "logging" is enabled.

Whenever this mode is selected, the **ResultXPos** and **ResultYPos** properties represent the center of the specified line and can be used to further propagate the vision tree.

<u>**Returning Grayscale Values**</u>

If the **ResultMode** is set to "RESULT_GRAYLEVEL", rather than computing edge points, this tool returns the grayscale (intensity) value for each pixel along the length of the specified line. Each grayscale value can range from 0 (black) to 255 (white).

If you are executing GPL version 2.0 or later, an array of up to 1500 pixel values can be fetched using the **VisResults Info** property. This array of results is also displayed in the Vision Results panel of the PV user interface in the "Special Results" section and can be recorded if "logging" is enabled.

Whenever this mode is selected, the **ResultXPos** and **ResultYPos** properties represent the center of the specified line and can be used to further propagate the vision tree.

Special Feature Buttons (located above the property editor)

Adjust Threshold

Pressing this button displays a pop-up window that permits the value of the **Edge Threshold** property to be manually adjusted while dynamically viewing the effect of the new setting in the Camera Display. If the **DesiredAlg** mode is "EDGE_BINARY", the display illustrates the pixels above the threshold as white. Otherwise, for the grayscale modes, the display indicates the edges that satisfy the threshold criterion in white.

Show Histogram

If "RESULT_HISTOGRAM" mode is selected, this button will pop-up a window that displays the histogram for the specified line in the current image. A sample graph is shown below. This graph corresponds to the tool placement described in the Example below. You will note the large number of white pixels with grayscale values around 238 and a distribution of dark pixels with grayscale values between 34 and 68.
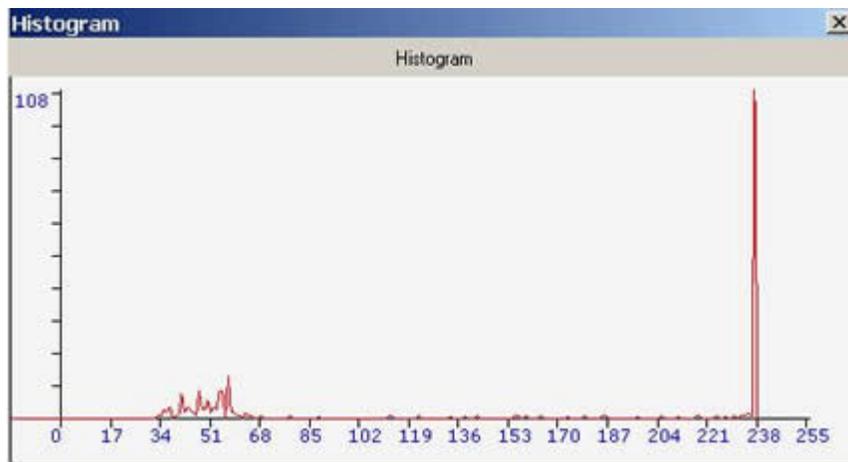
Show Gray Level

If "RESULT_GRAYLEVEL" mode is selected, this button will pop-up a window that displays the grayscale levels for each pixel along the specified line in the current image. A sample graph is shown below. This graph corresponds to the tool placement described in the Example

below. Note the low intensity values for the pixels that correspond to the black sections of the object.
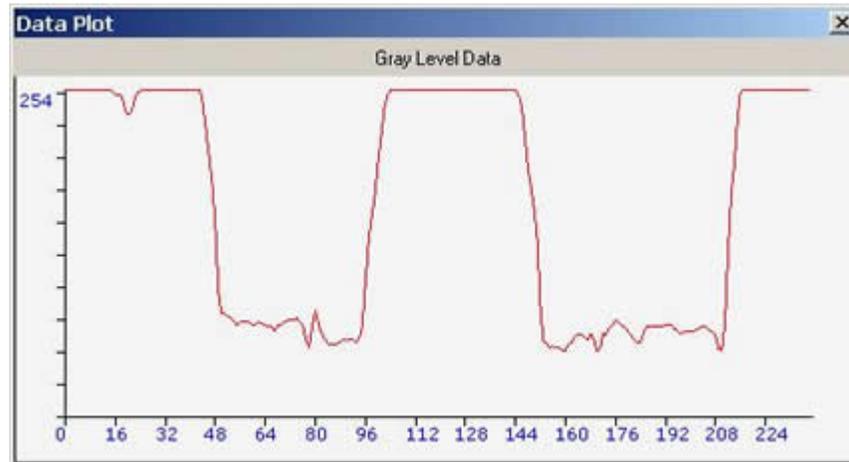
Examples

In the following example, an edge locator starts on the left of the image (at the square resizing handle) and processes moving to the right (towards the round rotation handle). In this case, four edges are detected and can be utilized to measure the diameter of the center hole and the thickness of the two sides. (Note, an Arc Fitter could have been used to determine the diameter of the center hole and would have resulted in a more accurate measurement.)
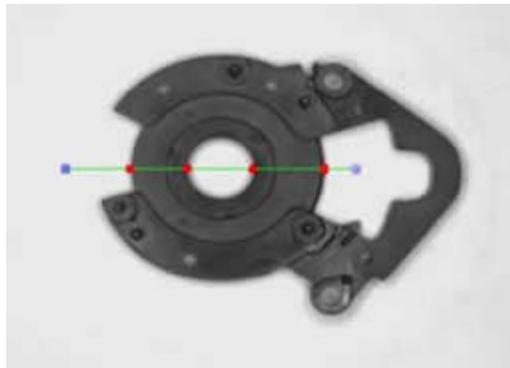
See Also

Vision Toolkit | **Arc Fitter** | **Line Fitter**

Finder   Tool

Most powerful vision tool that identifies randomly placed parts in a camera image and returns their position, orientation and scaling to within sub-pixel accuracy.
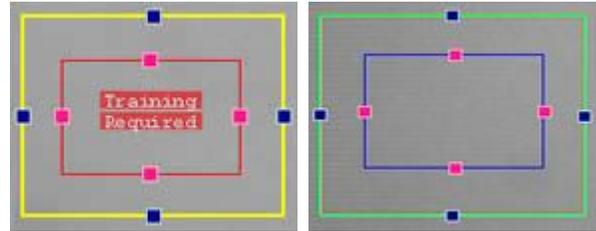
Prerequisites

    None

Properties: *Template "Training" Mode*

Property Name	Data Type	Range	Description
1. Identity			
Type	String	n/a	[Standard Identity properties](#)
2. Placement/Size			
Height	**Single**	0 - AOI (mm)	[Standard Placement / Size properties.](#) The X and Y values define the center of the rectangular template and the Height and Width define its dimensions.
Width	**Single**	0 - AOI (mm)	
XPos	**Single**	0 - AOI (mm)	
YPos	**Single**	0 - AOI (mm)	
3. Operation			
Camera	**Integer**	1 - 6	[Standard Operation property](#)
HiResMinScore	**Single**	0 - 1	When a template is compared against a

LoResMinScore	**Single**	0 - 1	region in a camera image, the "recognition score" indicates how well the two match. A value of 0 represents a total mismatch and a value of 1 represents a perfect match. The **HiResMinScore** and **LoResMinScore** properties dictate when a comparison is considered to represent a good match. The **LoResMinScore** is utilized at the lowest resolution (highest level of the search pyramid) to determine if a possible match is good enough to be further analyzed at the levels with higher resolution. The **HiResMinScore** dictates if a match is acceptable at the full camera image resolution and is acceptable as a final, good match. These values are set automatically during training and can be manually adjusted after training is completed. In general, the **LoResMinScore** is set equal to or lower than **HiResMinScore**. In fact, **LoResMinScore** can be set very low or even to 0. Since the **Finder** sorts all of the candidate matches at the lowest resolution and pursues best candidates first, setting **LoResMinScore** low just results in more candidates being pursued to the full resolution level. If a specific number of parts is being located, having additional candidates is not a problem since the finder will stop when the desired number of parts is located. However, if an unknown number of parts is being searched for, having this property set too low can result in excessive execution time. On the other hand, the setting for **HiResMinScore** is very important in terms of the quality of the **Finders** results. As soon as the system identifies the number of parts that you request whose recognition score satisfies this criterion, the **Finder** stops and returns these candidate matches. Setting this property higher results in higher quality matches being returned at the risk of rejecting acceptable matches.
MaxAngle	**Single**	-360 to 360	These properties dictate the range of

MinAngle	**Single**	-360 to 360	orientations that are tested relative to the trained orientation of the template. Normally, these properties are set to +/- 180 degrees to detect parts in any orientation. However, if a part can only rotate a limited amount relative to its taught orientation, reducing these values to reflect the smaller range of orientations will speed up processing. In addition, if the part has some type of orientation symmetry (e.g. a circle or a square), these values should be set to restrict the range of orientations. This speeds processing and prevents the robot from making unnecessary changes in orientation to grasp the part. These properties should be set as follows for the common types of symmetries: Circle: +/- 0 deg Square: +/- 45 deg Rectangle: +/- 90 deg Equilateral triangle: +/- 60 deg Hexagon: +/- 30 deg By default, these properties are set to +/- 180 degrees.
3A. Advanced Operation			
Accuracy	**Integer**	1 - 100	This property controls the amount of effort and time that is spent on performing the final refinement of the location and orientation of a matched template. Smaller values result in less accuracy but faster execution times. Values above 10 provide higher sub-pixel accuracy at the expense of execution times. By default, this property is set to 10.
Depth	**Integer**	1 - n	This is the maximum pyramid search level to build and use. At runtime, to optimize its execution, the **Finder** initially searches for matches between the template model and the camera image at reduced resolutions. Each level of reduced resolution is a factor of 2 smaller in width and height. During training, PreciseVision automatically computes the maximum number of levels to use based upon the loss of distinctive features and other criterion and sets this property value. After training, the **Depth** can be reduced to limit the number of reduced resolution levels considered. This is beneficial if the system is too aggressive and selects too many levels. The number of levels does not affect the accuracy of the final results of the **Finder** since the last match is always performed at

			the full image resolution.
GainLimit	**Single**		Each time the template is compared to a region of a camera image by the **Finder Tool**, to enhance robustness, the comparison is automatically normalized to eliminate differences due to the overall brightness of the image. In particular, each pixel value is effectively adjusted as follows: New_value = Gain*image_pixel + Offset This normalization process is controlled by the following properties:
OffsetLimit	**Integer**	0 - n	**GainLimit** restricts the range of the Gain. If non-zero, the "Gain" will be restricted to fall between 1/**GainLimit** and **GainLimit**. By default, this property is set to 4. **OffsetLimit** restricts the amount by which the overall brightness is adjusted up or down, i.e. this property specifies the maximum absolute value of the "Offset". By default, this property is set to 64.
MaxScale	**Double Array**		These properties can be set to locate parts that have a variable scaling (size). This can occur if the part can be at different distances from the camera while still in acceptable focus. Allowing scaling variations can be very useful, but it does require additional execution time. The first value in each of these arrays is for the X scaling and the second is for Y. By default, all four values are set to 1.0, which indicates no scaling is permitted.
MinScale	**Double Array**		
Speed	**Single**	-1 or 1 to 100	This property controls how quickly the tool executes at runtime versus how meticulously it attempts to find matches. If this value is set to 1, the Finder executes more slowly but with the highest reliability. If it is set to 100, it executes at the fastest speed but may fail under non-ideal conditions. During training, the value of this property is automatically set to a conservative value that achieves a good balance of speed and reliability. After training, the value can be modified.

Properties: *Runtime "Find" Mode*

Property Name	Data Type	Range	Description
1. Identity			
Name	**String**	n/a	[Standard Identity properties](#)

Type	String	n/a	
2. Placement/Size			
Height	**Single**	0 - AOI (mm)	[Standard Placement / Size properties.](#) The X and Y values define the center of the rectangular search region and the Height and Width define its dimensions.
Width	**Single**	0 - AOI (mm)	
XPos	**Single**	0 - AOI (mm)	
YPos	**Single**	0 - AOI (mm)	
RelAngle	**Single**	-360 to 360	
RelXPos	**Single**	0 - AOI (mm)	
RelYPos	**Single**	0 - AOI (mm)	
3. Operation			
Camera	**Integer**	1 - 6	[Standard Operation properties](#)
Relative-ToolName	**List**	n/a	
PrecentClipped	**Integer**	0 - 50	This property allows parts to be recognized even if the matching template area is partially outside of the search region. Normally, this property is set to 0 and the **Finder** only searches for X/Y template center positions that have the template fully within the search area. In this case, this tool will not return a match position where the center of the template is closer than 1/2 the template's width or height to a side of the search region. If the **PercentClipped** is set to 50%, all template center positions within the search area will be considered. As this property value is increased, the recognition match score that is compared to **HiResMinScore** and **LoResMinScore** will be reduced somewhat, but not in proportion to the value of this property. Also, this property only applies to template matches that protrude beyond the search space and does not deal with occlusion due to overlapping parts. To allow for overlapping parts, the **HiResMinScore** and **LoResMinScore** should be reduced. By default, this property is set to 0.
4. Inspection Settings			
InspectActual	Single		[Standard Inspection Settings properties](#)
InspectLabel	**List**	NONE / PASS_AND_FAIL / PASS / FAIL	

InspectType	**List**	NONE / RESULT_ANGLE / RESULT_SCORE / RESULT_ID	
InspectMax	**Single**		
InspectMin	**Single**		
InspectPassed	List	Pass / Fail	
5. Results Settings			
MaxResults	**Integer**	-1 or 1 to n	[Standard Results Settings properties](). **MaxResults** defines the number of matches to find each time that this tool is executed. This tool will execute more quickly if a known number of matches are expected and this property is set accordingly. This is due to the fact that execution terminates as soon as the expected number of successful matches is found. Until all expected matches are found, this tool will continue to exhaustively test all possible matches. By default, this property is set to -1, exhaustive matching.
OffsetAngle	**Single**	degrees	
OffsetX	**Single**	mm	
OffsetY	**Single**	mm	
ResultSelect	**Integer**	1 - n	
ShowResults	**List**	NONE / FRAME / LINE / POINT	
6. Results			
ResultCount	Integer	0 - n	[Standard Results properties]()
ResultErrorCode	Integer		
ResultAngle	Single	-360 to 360	
ResultXPos	Single	mm	
ResultYPos	Single	mm	
ResultScaleX	Single		These values indicate the X and Y scale factors that were computed when matching the template to the camera image. If scaling was not enabled, these values will be set to 1. *[GPL: VisResult.Info(1), VisResult.Info(2)]*
ResultScaleY	Single		
Recognition Time	Single	msec	Total time required for the last execution of this tool in milliseconds. *[GPL: VisResult.Info(3)]*
ResultScore	Single	0 - 1	This is the "recognition score" that indicates how well the template matched the located part. A value of 0 represents a total mismatch and a value of 1 represents a perfect match. Accepted scores will all be greater than or equal to the **HiResMinScore** property value for the associated template. *[GPL: VisResult.Info(0)]*

Remarks

The **Finder** is the most powerful tool in the toolkit and provides the most general method for identifying and locating parts in grayscale images. It can locate randomly placed parts with both simple and complex shapes and will return their part type, position, orientation and scale. There are other tools in the toolkit that are appropriate for specialized situations such as binary images or non-rotated parts. Also, for some simplified or constrained cases, it is relatively easy to combine **Line Fitters** and other tools to locate parts. However, the **Finder** can handle the widest range of parts and situations, has a very simple one-shot teaching method, and returns accurate sub-pixel results.

To address applications like locating a secondary object with a primary object, the **Finder** can be placed relative to the results of any other tool including another **Finder**. The one restriction is that the **Finder** search window is always orthogonally aligned with the camera frame of reference.

Typical uses for this tool include: finding fiducial marks on a printed circuit board, locating parts randomly placed on a conveyor belt and refining the position and orientation of sample trays.

Finders and Finder Templates

To teach the **Finder** the visual characteristics of a part, a single image of the part must be "trained". The training process consists of bounding the entire part plus some of the adjacent background with a red "Template rectangle". This red rectangle defines the "**Finder Template**" for the part. When PreciseVision subsequently searches for the part during runtime, it is looking for matches for the Template in the image. When the system reports the position and orientation of a matched part, it is actually stating the position of the center of the Template when the match was found and the amount by which the Template was rotated.

During the training process, a yellow "Training rectangle" must also be positioned around the Template. This yellow rectangle is used to teach the system how to distinguish the Template from other objects or background clutter that might be encountered at runtime. If different types of parts or clutter are expected to be present, the system will work best if one and only one full copy of the desired target part plus the other objects and the clutter are enclosed by the yellow rectangle. In general, the larger the yellow rectangle, the better the Finder will operate since this allows the Finder to try more positions and orientations for the template as it optimizes its search parameters. Please note that since PreciseVision is a 2D vision system, different poses (stable states) of the same part that produce different 2D images are considered different parts.

The following picture illustrates how the training rectangles should be positioned to teach the **Finder** how to locate one orientation of a part while ignoring a second stable state.
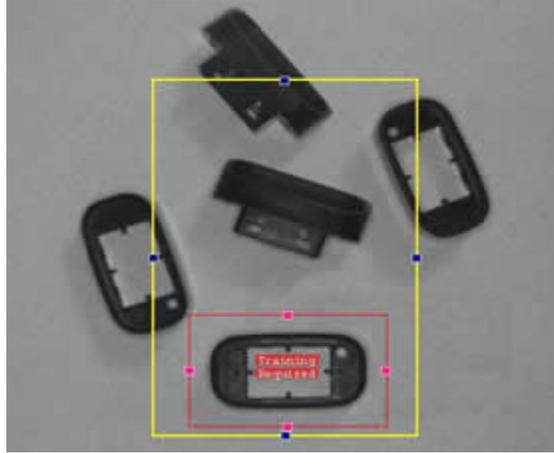
If two or more types of parts are to be identified and located in the same image, multiple **Finders** and their Templates should be trained and included in the Vision Process.

After the Template and Training rectangles have been properly positioned, the **Finder** can be taught by pressing the Train icon on the Vision Tool Definition tool bar. To simplify the training and setup process, when you create a new **Finder**, a wizard is automatically displayed that walks you through the training steps. Using this wizard, performing the training and setting up the key **Finder** properties is extremely simple.

Training and Finding Modes

The **Finder** is always in one of two modes: "Training" or "Finding". When this tool is first created, it is automatically placed into the Training mode. In this mode, the property list displays the size and the position of the Template and other properties that are specific to the type of part being taught, e.g. parameters for part symmetries, expected scale variations, etc. In the Vision Camera Display window, the yellow Training rectangle and the red Template rectangle are displayed.

After a Template has been trained, the tool will be in Find mode and an alternate property list is displayed. This property list presents the size and position of a green "Search rectangle". This rectangle defines the camera image area that the **Finder** will search for matches to the Template. The alternate property list also contains parameters that control the search, e.g. the maximum search time, number of parts to find, etc. In the Vision Camera Display Window, the green Search rectangle and the blue "taught Template rectangle" will be displayed.

To switch between Training and Finding modes, select a **Finder** and right click in the Vision Tool Definition Window or the Camera Display Window or select the appropriate icon in the Vision Tool Definition tool bar.

The Finder Search Algorithm

The heart of the **Finder Tool** is a new patented algorithm that combines an intelligent search technique with the robustness of normalized correlation to match the Template with regions in the search region. The intelligent search technique avoids the computational requirements of applying correlation to a multi-dimensional search that accommodates concurrent changes in position, orientation and scale. Also, this technique does not introduce the problems encountered with correlation optimization

methods such as skipping pixels, while still retaining the important benefits of a normalized correlation match.

Like most search methods, this tool employs reduced resolution images for the initial search passes in order to quickly identify candidate positions. This method is often called a pyramid search since each successive search level has half as many pixels in each row and column. During the training process, this tool automatically computes the maximum number of levels that can be utilized while still retaining sufficient distinctive features to differentiate the target object from the background and other objects.

At each position where the template (or a reduced resolution template) is compared to the image (or a reduced resolution image), a normalized comparison operation produces a "recognition match score". This score ranges from 0, which indicates no match, to 1, which indicates a perfect match. At the reduced resolutions, all of the locations whose score exceeds a minimum value are retained and ordered according to their score value. The tool then pursues each of these candidate locations and validates whether the location also satisfies a minimum match score when tested at the full camera resolution. When the system finds the requested number of successful results or all of the candidate locations have been exhausted, execution is terminated.

As a final step, if the Inspection Setting has been defined, each of the successful results will be tested against the Inspection criteria and will be tagged as to whether it "Passes" or "Fails".

Optimizing The Finder And Templates

The following are a number of guidelines for optimizing the performance of the **Finder** and **Finder Templates**.

Template (Red) Rectangle Guidelines

- The Template should fully enclose the target object and should contain no other objects or parts of objects.
- In general, the Template rectangle should be larger than the target object. It is best to have a boundary of at least 16 camera pixels all around the target object (more pixels are better than fewer pixels). For a 640x480 camera, 16 pixels is 1/30th of the full height of the camera image. This permits the system to capture the edges between the object and the background. This is especially important if the object does not have many internal features.

Good - partial border   Bad - no border
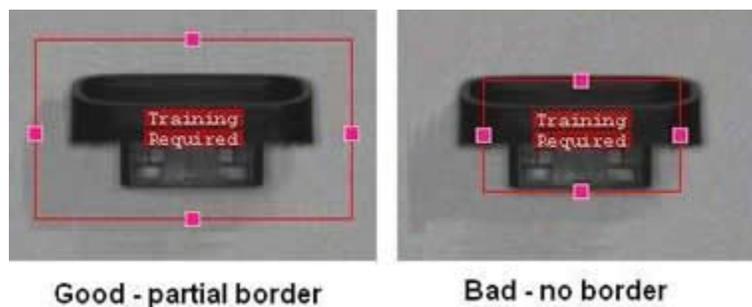
- If the background varies significantly (e.g., there is dirt and paint blotches on a conveyor belt), then the part should be placed on a uniform, neutral background to teach the Template.

- The target object should be oriented in the camera image in what you want to define as 'Zero' degrees. That is, when the Finder later locates a part is in the trained orientation, it will return an orientation angle of 0 degrees.

Training (Yellow) Rectangle Guidelines

- The Training rectangle must be larger than the Template and must fully enclose the Template.
- The Training rectangle should include one and only one full image of the target object. Since the Finder uses the contents of the Training rectangle to determine distinguishing features of the Template, a second complete copy of the Template will confuse the system and will generate an error. It is acceptable to have additional partial images of the target object in the Training rectangle so long as each is less than 50% of the target object (see picture above in the Template discussion for an example).
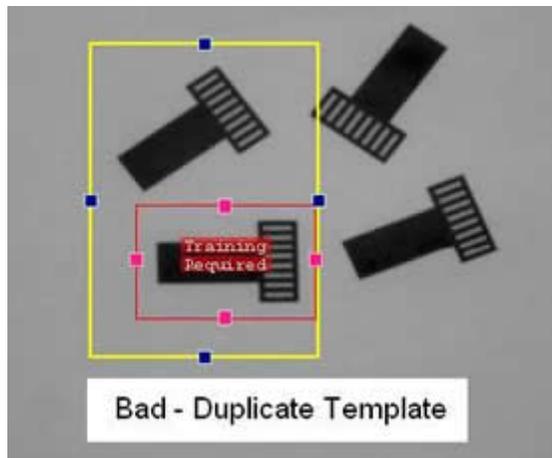
Bad - Duplicate Template

- This rectangle should include any other objects or clutter that are expected to be encountered during runtime to aid the system in learning to distinguish between the target object and its surroundings.
- This rectangle should be as large as possible without violating the rules above. A larger region permits the system to test more positions and orientations for the Template during the training phase and assists in optimizing the runtime search strategy.

Finder Properties

- If your part is symmetric, e.g. a circle, a square, a rectangle, set the **MinAngle** and **MaxAngle** properties to limit the rotation of the results.
- If any portion of the part might extend beyond the search window, set the **PrecentClipped** property appropriately.

System and Camera Setup

- Prior to training or executing this tool, calibrate the camera's pixel to millimeter relationship. <u>Failure to do so may result in parts not being located.</u> This is due to the fact that many cameras do not have square pixels, i.e. their X and Y scaling factors are different. If this difference in scaling is not accounted for properly, as parts are rotated, their shape will be distorted and their size will be incorrect.

- Larger target objects (and therefore larger templates) generally perform better than small templates for various reasons: finding a very small template in a large search space is comparable to finding a needle in a haystack; larger templates usually have more image features; larger templates usually can be found with higher accuracy particularly in orientation; and larger templates containing some big features can be searched using taller search pyramids thereby accelerating the search.

Optimal Target Objects

- Target objects that contain corners that form a 90-degree angle result in more reliable recognitions and higher accuracy.
- Sharp features in focus provide more accurate results than smooth or blurred features.
- If the template has many edges in one direction but only a few in the orthogonal direction, matched locations will have greater accuracy in the direction perpendicular to the majority of the edges. In the figures below, templates "A" and "B" contain many vertical edges so both can be located with high accuracy in the horizontal direction. But template "A" has very few horizontal edges compared to "B", so along the vertical direction, "B" will be found with higher accuracy than template "A".
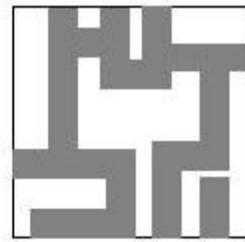
Template A (poor)   Template B (good)

- A wide dynamic range of brightness in the template provides more accurate results than an image with a narrow range. However, avoid saturation. Image information is lost when pixels are clipped at the two ends of the brightness range.
- Fine details in the image enhance the overall accuracy of the position calculations, but large features enhance the speed of search. Large features enable use of taller search pyramids.

Examples

In the following example, a **Finder Template** is trained on one of six identical parts. Since the parts are all the same, the yellow Training rectangle is only placed around one complete part although other partial parts (each less than 50% of the full object) are included in order to enlarge the Training rectangle. Also, for a reasonable template boundary and for an intuitive orientation angle, the horizontal part is trained rather than the parts at a skewed angle. After the Training and the Template rectangles are positioned, the Train icon can be pressed.

PreciseVision
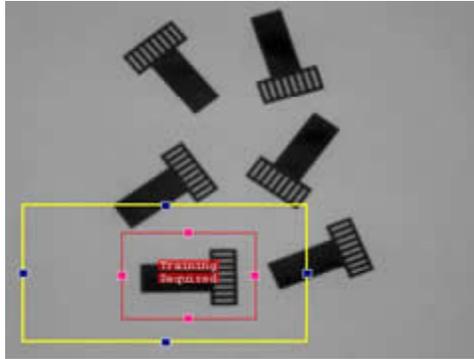
For this simple part, training completes in approximately one second after which the tool is automatically placed into Find mode and the search operation is executed once. At the conclusion of the search, all six parts will be identified and each of the part's position and orientation will be indicated by yellow coordinate axes as illustrated in the following picture.
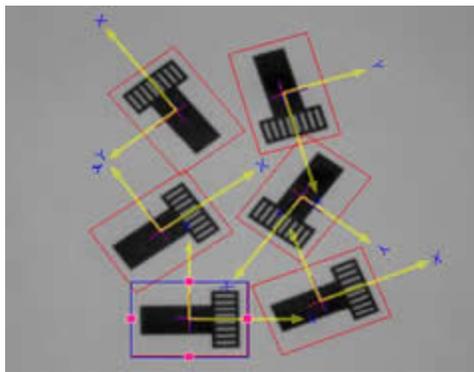
Once in Find mode, the "Execute Selected Tool" green circular icon is enabled on the Vision Tool Definition tool bar. For testing purposes, the **Finder** can be repeatedly executed by clicking this icon.

See Also

Vision Toolkit | Connectivity

Fixed Frame Tool

Computational tool that places a reference frame at a fixed image coordinate or at a constant offset relative to another vision object. Can optionally index the frame in an X and/or Y grid pattern to repeatedly execute any linked tools.

Prerequisites

None

Properties

Property Name	Data Type	Range	Description
1. Identity			
Name	**String**	n/a	Standard Identity properties
Type	String	n/a	
2. Placement/Size			
Angle	**Single**	-360 to 360	Standard Placement / Size properties. The X and Y values define the origin of the reference frame and the Angle specifies its rotation.
XPos	**Single**	0 - AOI (mm)	
YPos	**Single**	0 - AOI (mm)	
RelAngle	**Single**	-360 to 360	
RelXPos	**Single**	0 - AOI (mm)	
RelYPos	**Single**	0 - AOI (mm)	
3. Operation			
Camera	**Integer**	1 - 6	Standard Operation properties
Relative-ToolName	**List**	n/a	
3A. Advanced Operation			
FramePatternXIndex	**Integer**	1 - n	If the X and/or Y index values are greater than 1, the position of the Fixed Frame is indexed in a grid pattern the number of X and/or Y steps by the appropriate Pitch values. This permits any linked tools to be automatically and repeatedly executed in a grid pattern.
FramePatternYIndex	**Integer**	1 - n	
FramePatternXPitch	**Single**	mm	
FramePatternYPitch	**Single**	mm	
5. Results Settings			

PreciseVision

| ShowResults	List	NONE / FRAME / LINE / POINT	The frame can be graphically represented as a coordinate frame, line or a point.
6. Results			
ResultAngle	Single	-360 to 360	Standard Results properties
ResultXPos	Single	mm	
ResultYPos	Single	mm	

Remarks

A Fixed Frame can be used in several different ways. It can be defined relative to another vision object with a fixed position and angular offset. This is an alternative to directly setting an offset in the Results Settings of most tools and has the advantage of allowing the offset to be graphically specified by dragging the Fixed Frame in the vision window.

Alternately, a Fixed Frame can be placed in an image at a specified position and orientation and can then serve as a datum from which other measurements are made.

Lastly, if the Pattern properties are utilized, a Fixed Frame can be automatically indexed about an X and/or Y grid pattern. This has the benefit of repeatedly executing any tools linked to (i.e. defined Relative-to) the Fixed Frame in a grid pattern.

Examples

Below is an example of a Finder that returns a frame that is at the center of the part. A Fixed Frame is placed relative to the finder and used to effectively re-define the part's frame of reference. Additional vision tools can now be placed relative to the Fixed Frame. The advantage of doing this is that if the Finder template is re-trained it may not be placed in exactly the same position and orientation relative to the part. The Fixed Frame can then be adjusted to account for any difference in the Finder position and orientation. By using this relative reference frame, we can easily ensure that all subsequent tools will maintain their positional relationship with respect to key features of the part.
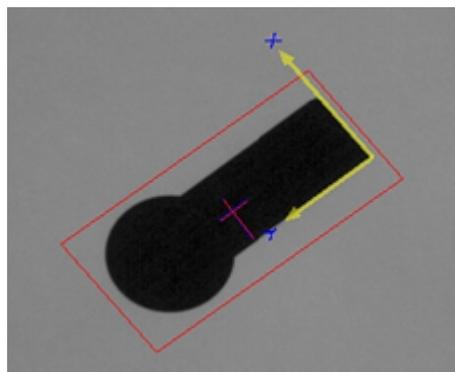

See Also

[Vision Toolkit](#) | [Line-Line Frame Tool](#) | [Point-Line Frame Tool](#) | [Point-Point Line Tool](#)

Line Fitter Tool

Vision tool that searches a specified region for edge points and returns the line that best fits the edges.

Prerequisites

> None

Properties

Property Name	Data Type	Range	Description
1. Identity			
Name	**String**	n/a	Standard Identity properties
Type	String	n/a	
2. Placement/Size			
Angle	**Single**	-360 to 360	Standard Placement / Size properties. The X and Y values define the center of the tool and the center of the search region. The Height and Width specify the size of the search region.
Height	**Single**	0 - AOI (mm)	
Width	**Single**	0 - AOI (mm)	
XPos	**Single**	0 - AOI (mm)	
YPos	**Single**	0 - AOI (mm)	
RelAngle	**Single**	-360 to 360	
RelXPos	**Single**	0 - AOI (mm)	
RelYPos	**Single**	0 - AOI (mm)	
3. Operation			
Camera	**Integer**	1 - 6	Standard Operation properties
Relative-ToolName	**List**	n/a	
EdgeMode	**List**	..._NEAREST_DARK / ..._NEAREST_NOMINAL / ..._NEAREST_LIGHT / ..._MAX_GRADIENTS / ..._MAX_CONCENTRATION	If multiple lines exist within the search region, this parameter specifies which line should be returned. In addition to selecting the line closest to the dark side, light side, nominal (center line), or greatest concentration of edges, you can also select the line with the

			highest contrast.
EdgeThreshold	**Integer**	0 - 254	Specifies the threshold below which weak (low contrast) gradients edges are ignored. The smaller the number, the more sensitive the system is in locating edges. Please see the description of the Edge Locator for more information on edge gradients. The default value is 30.
3A. Advanced Operation			
BiasFilter	**List**	..._BIAS_NONE / ..._BIAS_TO_LIGHT / ..._BIAS_TO_DARK	For lines with an irregular or jagged edge, this parameter can be set to provide extra weight (importance) to edges that are closest to the light or dark side of the search region. For example, for a saw toothed pattern, this property could be set to fit the line to the tips of the high or low points rather than along the center line.
MaxEdgePoints	**Integer**	2 - n	Maximum number of edge point searches to perform. Higher numbers produce greater accuracy at the expense of execution speed. The minimum value of this parameter is 2 and this value is automatically limited to the number of pixels across the width of the tool. The default value is 30.
MaxIterations	**Integer**	0 - n	Maximum number of iterations for filtering. The filtering algorithm repeatedly removes outlier points, refitting the line each time, until no more points need to be removed or the maximum number of iterations have occurred. A value of zero disables filtering. The default value is 5.
MinFilterDistance	**Single**	0 - n.nn	Absolute minimum filter distance in pixels. No edge points closer to the fitted line than this are discarded during the iterative process. A minimum distance is needed because the standard deviation of distances to the fitted line can be less than a pixel with a

			good image of a clean edge. The default value is 1.
SigmaFilter	**Single**	0 - n.nn	Filter width in units of standard deviations. This value is multiplied by the standard deviation of the edge points' distances to the fitted line to compute the distance threshold beyond which edge points are removed during the iterative fitting process. The default value is 1.5.
4. Inspection Settings			
InspectActual	*Single*		Standard Inspection Settings properties
InspectLabel	**List**	NONE / PASS_AND_FAIL / PASS / FAIL	
InspectType	**List**	NONE / ..._RMS / ..._ANGLE / ..._NUM_EDGES_FOUND / ..._NUM_EDGES_USED	
InspectMax	**Single**		
InspectMin	**Single**		
InspectPassed	*List*	*Pass / Fail*	
5. Results Settings			
OffsetAngle	**Single**	degrees	Standard Results Settings properties
OffsetX	**Single**	mm	
OffsetY	**Single**	mm	
ShowResults	**List**	NONE / FRAME / LINE / POINT	
6. Results			
ResultErrorCode	*Integer*		Standard Results properties
ResultAngle	*Single*	*-360 to 360*	
ResultXPos	*Single*	*mm*	
ResultYPos	*Single*	*mm*	
ResultNum-EdgesFound	**Integer**		Number of edge points found. *[GPL: VisResult.Info(1)]*
ResultNum-EdgesUsed	**Integer**		Number of edge points remaining after the filtering process. *[GPL: VisResult.Info(2)]*
ResultRMS	**Single**		Root mean square of the line fit. *[GPL: VisResult.Info(0)]*

Remarks

The Line Fitter searches the region defined by its green and blue outlined rectangle for edge points and fits a line to the edge points using a least squares technique. By

computing the edge positions to a sub-pixel accuracy and by employing multiple such edges in its computation, this tool is able to generate very accurate sub-pixel results.

This tool is very useful for accurately locating straight edges of objects. Once found, these fit lines can be used to compute distances between important features or for generating reference frames that can accurately locate an object or features in an object.

In order to make this tool more discriminating, the Line Fitter only utilizes edges whose dark and light sides have roughly the same orientation as the dark (blue) and light (green) sides of the search area. So, when positioning this tool, you should place the center (nominal) line of the search region at approximately the position and orientation that you expect to find the line with the blue side on the anticipated dark side of the object.

The operation of this tool is basically performed in two steps. First, all of the edges are extracted in the search region. Secondly, a straight line is fit to the edges. Of the two operations, the tool spends most of its time extracting the edges. So, the speed and the accuracy of the Line Fitter can be traded-off by adjusting **MaxEdgePoints**. This property determines how densely the tool scans to detect edges. This parameter must be set to at least 2 in order to detect the minimum number of edges to define a line. At most, one edge search is performed for each pixel along the width of the Fitter. Execution time increases roughly in proportion to the value of **MaxEdgePoints**. On the other hand, the accuracy of the tools increases approximately as the square root of **MaxEdgePoints**. That is, if you increase **MaxEdgePoints** by a factor of 4, the tool execution time will quadruple but the accuracy will only double.

Once the edges are found, the Line Fitter will optionally perform an iterative fitting process to increase the robustness of the results. During each pass, edge points that are too distant from the line (outliers) are discarded. This filters out edges that are not a part of the line and whose inclusion would incorrectly offset the result. **MaxIterations** specifies the maximum number of fits to perform. After each line fit, edges that are beyond **SigmaFilter** standard deviations are rejected, unless the edge is within **MinFilterDistance** pixels of the fit line. The iterative process stops if (1) **MaxIterations** are performed, (2) no further edges are discarded, or (3) only two edge points remain.

At the conclusion of this tool, the center position and orientation of the best fit line are returned along with statistical data on the number of edges finally used and the root mean square of the line fit, which indicates the quality of the final fit. This statistical data can be tested via the "Inspect Settings" to yield a pass/fail indication for the operation.

Special Feature Buttons (located above the property editor)

Adjust Threshold

Pressing this button displays a pop-up window that permits the value of the **Edge Threshold** property to be manually adjusted while dynamically viewing the effect of the new setting in the Camera Display. The display indicates the edges that satisfy the threshold criterion in white.

Examples

In the following example, two Line Fitters are utilized to accurately determine the positions and orientations of the straight sides of an object. A Computed Line-Line Intersection tool then takes the output of the two Fitters and computes a reference frame

whose origin is at the intersection of the two fit lines. Since the two Line Fitters compute their results based upon multiple pixel values, their output values are very accurate.
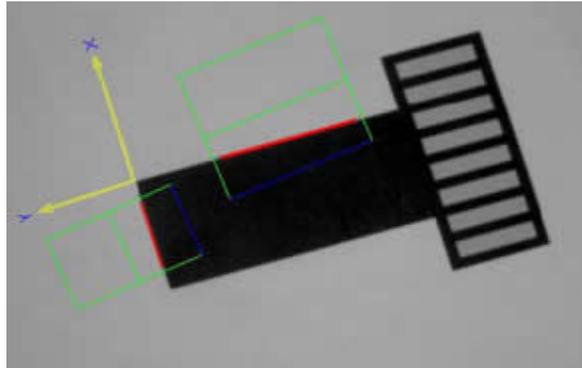

See Also

Vision Toolkit | **Arc Fitter** | **Edge Locator** | **Point-Point Line Tool**

Line-Line Frame Tool

Computational tool that determines the intersection of two lines defined by vision objects and returns a reference frame.
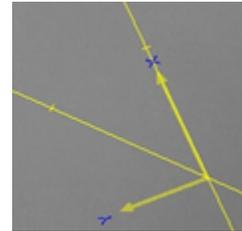
Prerequisites

Requires two vision objects that each supply either a line or a frame.

Properties

Property Name	Data Type	Range	Description
1. Identity			
Name	**String**	n/a	**Standard Identity properties**
Type	String	n/a	
3. Operation			
Camera	**Integer**	1 - 6	**Standard Operation property**
SourceLine1Name	**String**	n/a	Name of the vision object that defines the line that determines the X-axis of the computed reference frame.
SourceLine2Name	**String**	n/a	Name of the vision object that defines the second line that is intersected with the first line.
4. Inspection Settings			
InspectActual	Single		
InspectLabel	**List**	NONE / PASS_AND_FAIL / PASS / FAIL	**Standard Inspection Settings properties.** Can be used to validate the **ResultAngle**.
InspectType	**List**	NONE / RESULT_ANGLE	
InspectMax	**Single**		
InspectMin	**Single**		
InspectPassed	List	Pass / Fail	
5. Results Settings			
OffsetAngle	**Single**	degrees	**Standard Results Settings properties**
OffsetX	**Single**	mm	

| OffsetY	Single	mm	
ShowResults	List	NONE / FRAME / LINE / POINT	
6. Results			
ResultAngle	Single	-360 to 360	Standard Results properties
ResultXPos	Single	mm	
ResultYPos	Single	mm	

Remarks

This computational tool combines the results of two vision tools to compute a new reference frame. The inputs to this method from the vision objects are two non-parallel lines. The X-axis of the new reference frame will be collinear with the first input line. The origin of the new frame will be at the point where the two lines intersect.

This method is often used to uniquely define the position and orientation of a simple object or a section of an object. The resulting reference frame can then be used to position additional vision tools.

Alternately, this method can be utilized to compute the position of a corner of an object as determined by its two adjacent straight sides.

Examples

The following example illustrates how two Line Fitters can be applied to accurately identify the position and orientation of two adjacent sides of an object. The results of the two Fitters are then fed into a Line-Line Frame Tool to yield a reference frame that accurately represents the position and orientation of the enclosed corner.
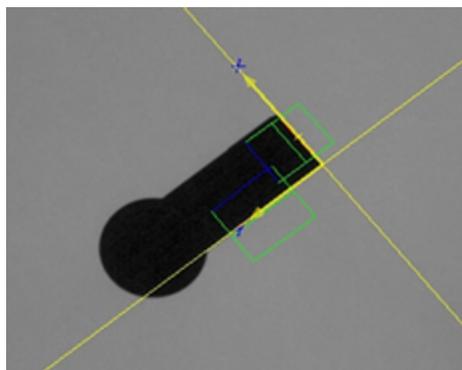

See Also

Vision Toolkit | **Fixed Frame Tool** | **Point-Line Frame Tool** | **Point-Point Line Tool**

Pixel Window Tool

Vision tool that counts pixels of a specific type or computes pixel statistics within a rotated rectangular or circular region. This tool is typically used to quickly determine if a feature is present or to verify that the average gray-level reading of a region is correct.

Prerequisites

None

Properties

Property Name	Data Type	Range	Description
1. Identity			
Name	**String**	n/a	Standard Identity properties
Type	String	n/a	
2. Placement/Size			
Angle	**Single**	-360 to 360	Standard Placement / Size properties. The X and Y values define the center of the region to be processed. The Height and Width define the dimensions of the region.
Height	**Single**	0 - AOI (mm)	
Width	**Single**	0 - AOI (mm)	
XPos	**Single**	0 - AOI (mm)	
YPos	**Single**	0 - AOI (mm)	
RelAngle	**Single**	-360 to 360	
RelXPos	**Single**	0 - AOI (mm)	
RelYPos	**Single**	0 - AOI (mm)	
3. Operation			
Camera	**Integer**	1 - 6	Standard Operation properties
Relative-ToolName	**List**	n/a	
PixelMode	**LIST**	BINARY_STATS / GRAY_STATES / EDGE_LEVEL	When this tool executes, it operates in one of three different modes: "BINARY_STATS" counts white/black pixels after thresholding. "GRAY_STATS" returns gray-level statistics (i.e. mean gray-levels).

			"EDGE_LEVEL" counts edge pixels after a gradient operator is applied. Where a property is only relevant for a specific mode, the property is labeled as BINARY, GRAY or EDGE.
CountWhitePixels	**Boolean**	TRUE / FALSE	(BINARY) If TRUE indicates that the white pixels are to be counted. Otherwise, black pixels are counted.
Threshold	**Integer**	0 - 254	(BINARY) This is the value used to convert gray scale values into black and white pixels. (GRAY) This value is ignored. (EDGE) Specifies the threshold below which weak (low contrast) gradient edges are ignored. The lower this property is set, the more sensitive the system is in locating edges. Please see the description of the [Edge Locator](#) for more information on edge gradients. The default value is 30.
ToolShapeType	**LIST**	RECTANGULAR / ROUND / SQUARE	Shape of the window in which the pixels are analyzed. A square window has its aspect ratio forced to 1:1.
4. Inspection Settings			
InspectActual	Single		
InspectLabel	**List**	NONE / PASS_AND_FAIL / PASS / FAIL	[Standard Inspection Settings properties](#)
InspectType	**List**	NONE / MEAN / NUM_PIXELS / STANDARD_DEV / PIXELSUM / PIXELSUMSQUARE	
InspectMax	**Single**		
InspectMin	**Single**		
InspectPassed	List	Pass / Fail	
5. Results Settings			
OffsetAngle	**Single**	degrees	[Standard Results Settings properties](#)
OffsetX	**Single**	mm	
OffsetY	**Single**	mm	

ShowResults	List	NONE / FRAME / LINE / POINT	
6. Results			
ResultErrorCode	Integer		Standard Results properties
ResultAngle	Single	0 (non-rotating)	(BINARY,EDGE) X and Y position represent the centroid of the counted binary or edge pixels.
ResultXPos	Single	mm	
ResultYPos	Single	mm	
ResultPixelCount	**Integer**		(BINARY,EDGE) Total number of counted binary or edge pixels in the rectangular region. *[GPL: VisResult.Info(0)]*
ResultMean	**Single**		(GRAY) Mean gray-scale value of all pixels in the region. *[GPL: VisResult.Info(1)]*
ResultPixelSum	**Integer**		(GRAY) Sum of gray-scale values of all pixels in the region. *[GPL: VisResult.Info(5)]*
ResultPixelSumSqr	**Integer**		(GRAY) Sum of the squares of the gray-scale values for all pixels in the region. *[GPL: VisResult.Info(6)]*
ResultStandardDev	**Single**		(GRAY) Standard Deviation of the gray-scale values for all pixels in the region. *[GPL: VisResult.Info(2)]*

Remarks

This tool very quickly processes a rectangular or circular region of a gray-scale image and either counts black or white pixels after thresholding, counts edge pixels after applying a gradient operator or collects image statistics, such as mean and standard deviation of the pixel gray-scale values.

This tool provides a very efficient means for quickly extracting gross statistical information about a rectangular or circular region. This tool is often used to quickly verify the presence of a feature or obstacle or to collect gray-scale statistics. For example, this tool is often employed to inspect a part to ensure that expected holes are present or that tabs have been removed. The BINARY mode can be applied to detecting defects in a uniform area such as inspecting for scratches or pits in a lens. The GRAY mode can also be used as a simple light meter to adjust the threshold for other operations or can determine the gray-scale uniformity of the region.

When this tool is operated in the BINARY mode, the specified region is converted to white and black pixels based upon the **Threshold** property. Then, either the white or the black pixels are counted based upon the **CountWhitePixels** property. The number of counted pixels and the centroid of all of the positions of the counted pixels are returned in the results.

If this tool is operated in the EDGE mode, the specified region is converted to edge pixels using a gradient operator and the setting of the **Threshold** property to select only strong edges. Then, the edge pixels are counted. The number of edge pixels and the centroid of all of their positions are returned in the results.

If this tool is operated in GRAY mode, statistics are collected on all of the pixel gray-scale values within the specified region. The returned results include the mean value, sum of the squares, and standard deviation.

Special Feature Buttons (located above the property editor)

Adjust Threshold

Pressing this button displays a pop-up window that permits the value of the **Threshold** property to be easily adjusted while dynamically viewing the effect of the new setting in the Camera Display. In BINARY mode, the displayed image shows all pixels converted to black or white, where the white pixels are the ones that are counted. In EDGE mode, the edge pixels that are extracted and will be counted are displayed in white.

See Also

Vision Toolkit | Color Pixel Window

Point-Line Frame Tool

Computational tool that takes a point and a line from two vision objects and returns a reference frame.
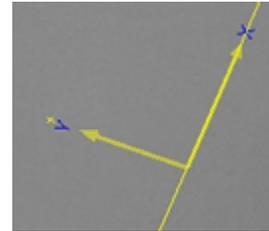
Prerequisites

Requires two vision objects, one of which supplies a line or a frame and the other of which supplies a point.

Properties

Property Name	Data Type	Range	Description
1. Identity			
Name	**String**	n/a	Standard Identity properties
Type	String	n/a	
3. Operation			
Camera	**Integer**	1 - 6	Standard Operation property
SourceLineName	**String**	n/a	Name of the vision object that defines the line.
SourcePointName	**String**	n/a	Name of the vision object that defines the point.
4. Inspection Settings			
InspectActual	Single		Standard Inspection Settings properties. Can be used to validate the **ResultAngle**.
InspectLabel	**List**	NONE / PASS_AND_FAIL / PASS / FAIL	
InspectType	**List**	NONE / RESULT_ANGLE	
InspectMax	**Single**		
InspectMin	**Single**		
InspectPassed	List	Pass / Fail	
5. Results Settings			
OffsetAngle	**Single**	degrees	Standard Results Settings properties
OffsetX	**Single**	mm	

| OffsetY	Single	mm	
ShowResults	List	NONE / FRAME / LINE / POINT	
6. Results			
ResultAngle	Single	-360 to 360	
ResultXPos	Single	mm	[Standard Results properties](#)
ResultYPos	Single	mm	

Remarks

This computational tool combines the results of two vision tools to compute a new reference frame. The inputs to this method from the vision objects are a line and a point that is not on the line. The X-axis of the new reference frame will be collinear with the input line. The Y-axis of the new frame will pass through the specified point.

This method is often used to uniquely define the position and orientation of a simple object or a section of an object. The resulting reference frame can then be used to position additional vision tools.

Alternately, this method can be utilized to compute the distance between a point and a line since the origin of the generated reference frame defines the position on the line that is closest to the input point.

Examples

The following example illustrates how a Line Fitter and an Edge Locator can be applied to accurately identify the position and orientation of one side of an object and a point on a perpendicular side. The results of the Fitter and Locator are then fed into a Point-Line Frame Tool to yield a reference frame that accurately represents the position and orientation of this section of the object. The origin of this reference frame also specifies the point on the line that is closest to the point determined by the Edge Locator.
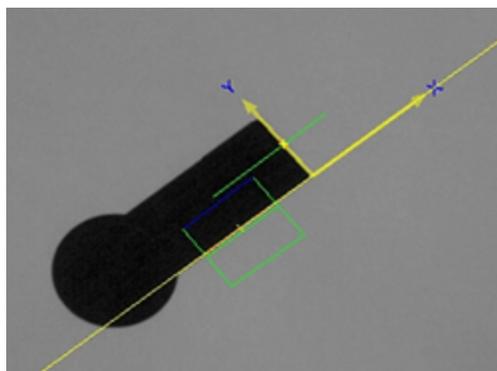

See Also

[Vision Toolkit](#) | [Fixed Frame Tool](#) | [Line-Line Frame Tool](#) | [Point-Point Line Tool](#)

Point-Point Line Tool

Computational tool that computes a line given two points from vision objects.
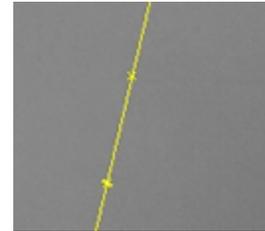
Prerequisites

Requires two vision objects that each supply either a point, a line or a frame.

Properties

Property Name	Data Type	Range	Description
1. Identity			
Name	**String**	n/a	Standard Identity properties
Type	String	n/a	
3. Operation			
Camera	**Integer**	1 - 6	Standard Operation property
SourcePoint1Name	**String**	n/a	Name of the vision object that defines one of the two points on the line.
SourcePoint2Name	**String**	n/a	Name of the vision object that defines one of the two points on the line.
4. Inspection Settings			
InspectActual	Single		
InspectLabel	**List**	NONE / PASS_AND_FAIL / PASS / FAIL	Standard Inspection Settings properties. Can be used to validate the **ResultAngle**.
InspectType	**List**	NONE / RESULT_ANGLE	
InspectMax	**Single**		
InspectMin	**Single**		
InspectPassed	List	Pass / Fail	
5. Results Settings			
OffsetAngle	**Single**	degrees	Standard Results Settings properties
OffsetX	**Single**	mm	
OffsetY	**Single**	mm	

| ShowResults	List	NONE / FRAME / LINE / POINT	
6. Results			
ResultAngle	Single	-360 to 360	
ResultXPos	Single	mm	[Standard Results properties](#)
ResultYPos	Single	mm	

Remarks

This computational tool combines the results of two vision tools to compute a line. The inputs to this method from the vision objects are two points. The resulting line will intercept each of the two input lines.

Once computed, the resulting line can be combined with other vision results to establish reference frames or as a datum for measurements.

Examples

The following example illustrates how two Edge Locators can be applied to find two reference points on the edge of an object to sub-pixel accuracy. The results of the two Locators can then be fed into the Point-Point Line Tool to compute a line that represents the location of one side of the object. This line can then be combined with other tools to measure the dimensions of the object or to establish reference frames. Note that, in general, a more accurate line fit can often be obtained using the Line Fitter Tool. However, there are incidences where an edge is non-uniform and selecting two good points to construct a line is advantageous.
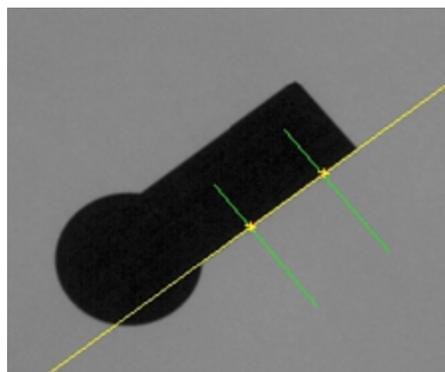

See Also

[Vision Toolkit](#) | [Fixed Frame Tool](#) | [Line-Line Frame Tool](#) | [Point-Line Frame Tool](#) | [Line Fitter Tool](#)

Sensor Window Tool

Vision tool that detects part movement or no movement within a specified region of the image. This tool takes multiple camera images in succession using a single camera and determines if there are any significant changes.

Prerequisites

None

Properties

Property Name	Data Type	Range	Description
1. Identity			
Name	**String**	an/a	Standard Identity properties
Type	String	n/a	
2. Placement/Size			
Height	**Single**	0 - AOI (mm)	Standard Placement / Size properties. The X and Y values define the center of the tool. The Height and Width specify the size of the image region being tested.
Width	**Single**	0 - AOI (mm)	
XPos	**Single**	0 - AOI (mm)	
YPos	**Single**	0 - AOI (mm)	
3. Operation			
Camera	**Integer**	1 - 6	Standard Operation property
DelayAfterComplete	**Single**	0-n	Specifies an amount of time to wait after the operation is completed (post-processing dwell time).
EdgeThreshold	**Integer**	0 - 254	Specifies the threshold below which weak (low contrast) gradients edges are ignored. The lower this property is set, the more sensitive the system is in locating edges. Please see the description in the Edge Locator Tool for more information on edge gradients. The default value is 30.
MaximumTimeout	**Single**	0-n (seconds)	Specifies a maximum time (in

124

			seconds) that the tool will execute. If this time is exceeded, the tool stops processing and returns an error.
PixelChangeAmount	**Integer**	0-n	If more than this specified number of edge pixels change between successive images, this tool declares that motion has been detected.
SenseMode	**List**	WAIT_MOTION_STOP / WAIT_MOTION_START	Specifies the operating mode of this tool. The default is to "wait for no motion".
3A. Advanced Operation			
ExposureTime	**Integer**		Same properties as in the Acquisition Tool. See that tool for details.
VideoGain	**Integer**		
VideoOffset	**Integer**		
5. Results Settings			
ShowResults	**List**	NONE / FRAME / LINE / POINT	Standard Results Settings property
6. Results			
ResultErrorCode	Integer		Standard Results properties
ResultAngle	Single	-360 to 360	
ResultXPos	Single	mm	
ResultYPos	Single	mm	
ResultActualTime	**Single**	sec	Actual time in seconds for operation to complete. *[GPL: VisResult.Info(1)]*
ResultPixelCount	**Integer**		Actual number of pixels detected during the operation. *[GPL: VisResult.Info(0)]*

Remarks

This motion sensing tool acquires a series of pictures sequentially and compares successive images to detect how many edges are changing. Based upon the number of changed edges, this tool decides upon whether any movement has been detected.

This tool determines movement by performing the following operations. First, it acquires a new image. Then the new image is subtracted from the previous image. A cross gradient edge extraction algorithm is performed across the resulting image and the edges are thresholded to produce a binary image. The binary image is eroded to eliminate insignificant changes. The resulting binary edges are counted and compared to the **PixelChangeAmount** to determine is sufficient pixels have changed to declare that movement has been detected. Based upon the setting of the **SenseMode** property, this tool either takes another picture and repeats the process or it terminates.

This tool requires successive images so a minimum of two pictures are always taken. During this operation, no graphics are displayed on the screen. A dialog is placed above

the vision window to indicate that the operation is in progress and permits the operator to abort the tool.

This tool can be applied in applications where parts may be falling or moving (e.g. rolling parts). Instead of applying a worst case fixed delay time, the Sensor Window Tool can be used to ensure the parts are not moving before locating or executing the remaining processes.

Special Feature Buttons (located above the property list):

Show Motion Image

Displays the output of comparing a single pair of images. Any pixels that are detected as changed (moving) are drawn in white. If zero motion is taking place, the entire image will be black.

Test Motion (Continuous)

Places tool in a continuous execution mode that allows the user to see the results of the motion dynamically. A small dialog box will appear in the center of the vision window that displays the number of pixel counts sensed as being in motion. Click the 'abort' button on the dialog to stop the test operation.

See Also

Vision Toolkit | Acquisition Tool | Edge Locator Tool

Tool Filter Tool

Filter tool that takes the output from another vision object that generates multiple sets of results, and returns a subset of the results based upon specified criteria.

Prerequisites

Requires another vision object that generates multiple sets of results.

Properties

Property Name	Data Type	Range	Description
1. Identity			
Name	**String**	n/a	Standard Identity properties
Type	String	n/a	
3. Operation			
Camera	**Integer**	1 - 6	Standard Operation property
SourceToolName	**String**		Name of the vision object that outputs multiple sets of results. These sets of results serve as the input to this tool.
3A. Advanced Operation			
SortResults	**List**	NONE / SORTFROMLEFT / SORTFROMRIGHT / SORTFROMTOP / SORTFROMBOTTOM	Automatically sorts the results based on the selection. This is useful for items on a conveyor belt that may need to be picked in order from the front to back of the conveyor.
ZeroAngle	**Boolean**	True / False	If True, will force the **ResultAngle** to ZERO. This is useful for randomly oriented parts that require the angle to be removed for picking. This operation does NOT change the sorting operation.
4. Inspection Settings			
InspectActual	Single		Standard Inspection Settings properties. All of the sets of results generated by the **SourceToolName** object are tested against this inspection criteria and those that pass become the output of the Tool Filter.
InspectType	**List**	NONE / RESULT_ID / RESULT_X_POS / RESULT_Y_POS / RESULT_ANGLE / RESULT_INSPECT_-STATUS	
InspectMax	**Single**		
InspectMin	**Single**		

InspectPassed	List	Pass / Fail	
InspectString	**String**	n/a	Optional label that is displayed in the vision window on all input results that satisfy the inspection criteria. This provides a convenient way to display part ID or other information.
5. Results Settings			
MaxResults	**Integer**	-1 or 1 to n	
OffsetAngle	**Single**	degrees	
OffsetX	**Single**	mm	
OffsetY	**Single**	mm	Standard Results Settings properties
ResultSelect	**Integer**	1 - n	
ShowResults	**List**	NONE / FRAME / LINE / POINT	
6. Results			
ResultCount	Integer	0 - n	
ResultAngle	Single	-360 to 360	Standard Results properties
ResultXPos	Single	mm	
ResultYPos	Single	mm	

Remarks

This tool inputs the results from another vision object that generated multiple sets of results, and applies criteria specified by the "Inspection Settings" to filter the results. This allows a process to isolate results that require further special processing or to create queues of results based upon specified criteria. Multiple **Tool Filters** can be applied to the same vision object to allow all of the results of the input tool to be segmented and processed.

For example, if the input tool is a **Finder**, special processing can be applied to all objects located by the **Finder** that are in a specific region of the screen by defining a **Tool Filter** with the **InspectType** set to "RESULT_X_POS" or "RESULT_Y_POS".

Examples

Below is an example of a single **Finder** that has located 6 identical parts. You will note that the bottom left part was used as the tool template, so that all future part orientations will be with respect to this horizontal position with the top of the "T" pointed to the right.
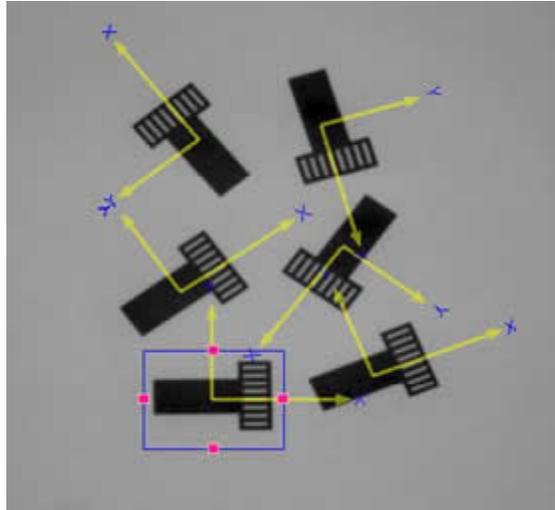
If we now wish to segment these 6 parts by their orientation, we can apply a **Tool Filter** that references the output of the **Finder** by specifying the **Finder**'s name as the **Filter's SourceToolName**. By setting the **Tool Filter's InspectType** to "RESULT_ANGLE", we can select only those parts with a specific orientation. If we set the **InspectMin** and **InspectMax** to -90 degrees and 90 degrees respectively, only those objects that are approximately pointed to the right will be selected as the output of the Filter. We can also add a label to these parts by filling in the **InspectString**. The picture below shows the results of this filter where **InspectString** has been set to "Pointed Right". You will note that only four of the six parts located by the Finder will be output by the Tool Filter. If we wanted to further process these four parts, the Tool Filter output could be utilized as the input to yet another tool.
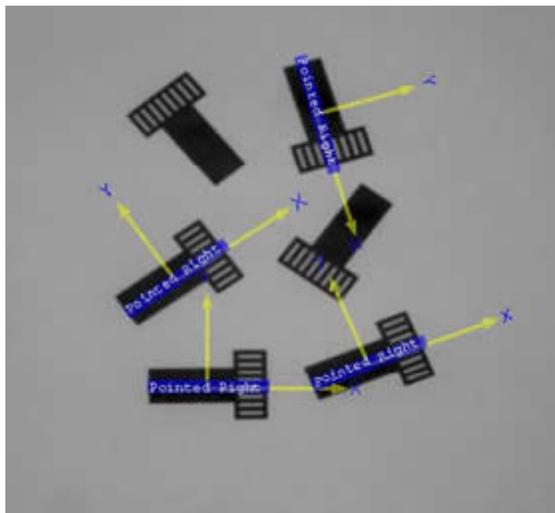
See Also

Vision Toolkit | **Finder Tool**

Camera Calibration

Camera Calibration Introduction

The PreciseVision camera calibration procedures generate the data necessary to convert camera pixel positions into real world units of measure, typically millimeters. For robot guidance, this calibration also translates camera positions into the robot's world coordinate system. This permits a robot to grasp a part that has been located by the vision system. Consequently, a camera must be calibrated with respect to a robot before a vision-guided application can be executed. Even for simple vision testing where PreciseVision is not communicating with a robot, calibration is still highly desirable (but not required), since the calibration process corrects for non-square pixels and cameras that are not perfectly perpendicular to the imaged plane, which causes perspective distortion. This correction is important as parts rotate to ensure that dimensions are always computed the same independent of the part's orientation.

So, when a camera is first put into use, it should be calibrated. In addition, calibration data should be updated if a camera with different optical characteristics is utilized, a camera is moved, a different camera lens is used or the length to the focal plane is altered. In general, it is always best to calibrate each camera that is placed into service and whenever any of the optical characteristics of a camera are modified.

The following table summarizes the camera calibration methods that are included with PreciseVision.

| Method	Description
Area Camera Calibration	Simple area camera calibration based solely on a standard target sheet. Does not require a robot and cannot be used for robot guidance.
Robot Vision Area Camera Calibration	Area camera calibration for use in robot guidance applications. Permits the position and orientation of part features in the camera's field of view to be converted into positions in the robot's world coordinate system.

In the next section, general instructions and information that apply to all calibration methods are presented. This is followed by detailed procedures for executing each of the specific calibration methods.

General Calibration Instructions

This section includes descriptions of general procedures that apply to all calibration methods. This section should be read prior to executing any of the calibration methods.

Standard Calibration Target Sheet

All calibration methods are required to establish the conversion between camera pixel positions and millimeters. In most cases, PreciseVision determines this relationship based on viewing a Standard Calibration Target Sheet that contains patterns of squares that have a known size and pitch. Three different pattern sizes are provided to accommodate different camera fields of view. The target sheet can be used without a robot or any automatic motion. This sheet looks like the following:
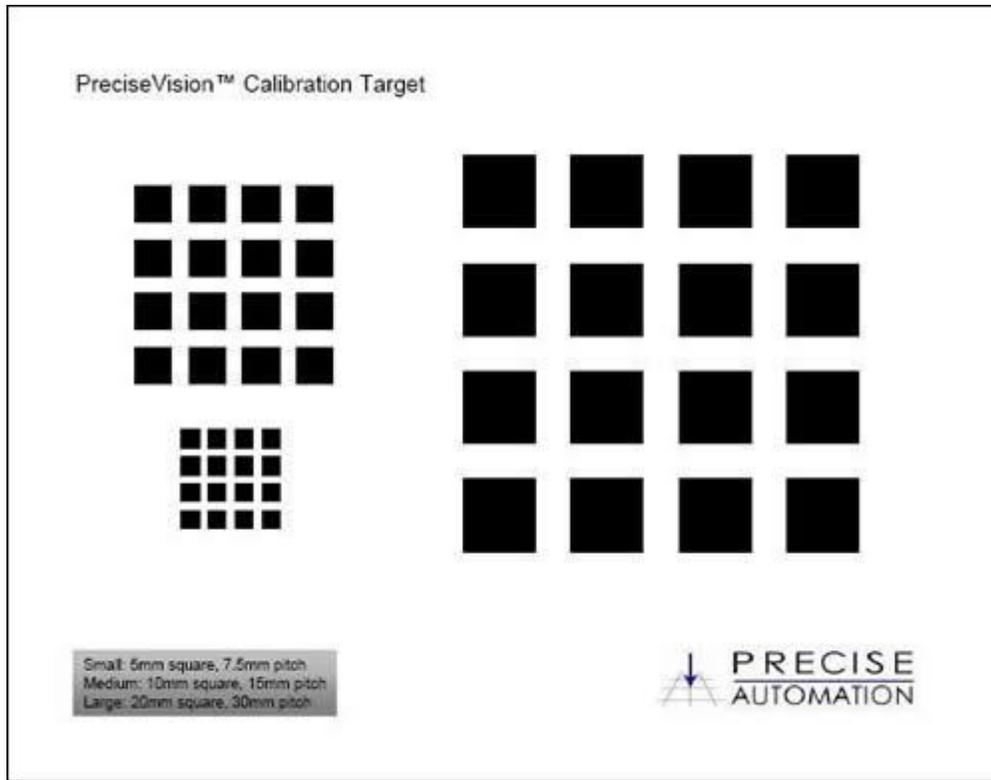
This target sheet is provided as a .pdf file and can be found via the Windows Start menu:

> ***Start > Programs > Precise Automation > PV xxx > Documentation > PreciseVisionCalibrationTarget.pdf***

or by browsing to the "\resource\" folder created upon installation of PreciseVision.

> **\Program Files\Precise Automation\PreciseVision #.#\resource\PreciseVisionCalibrationTarget.pdf**

The calibration target sheet should be printed out at the highest possible resolution and quality. When you are instructed to do so, this sheet should be placed in the focal plane of the camera. The focal plane of the camera should be as close as possible to the actual Z height of the primary part features to be located or inspected.

Camera Setup

Prior to executing a calibration process, the camera must be rigidly mounted, selected in software and carefully setup. The camera can be selected from the ***View > Select Camera*** top-level menu. The aperture and focus can then be manually set by viewing the camera image in "live video" mode. If the camera is not in this mode, it can be initiated by pressing an icon in the main toolbar. Finally, the camera

gain and offset that control the brightness and brightness range should be set by selecting **View >
Camera Gain and Offset**.

Calibration Pop-up Dialog Panel

Each of the calibration methods is initiated from the top menu bar by making a selection under **Tool >
Calibrate Vision.** When a method is started, it will display a pop-up dialog panel that is customized for
the selected method. However, each of these panels has several fields and buttons in common. A
typical panel looks like the following:
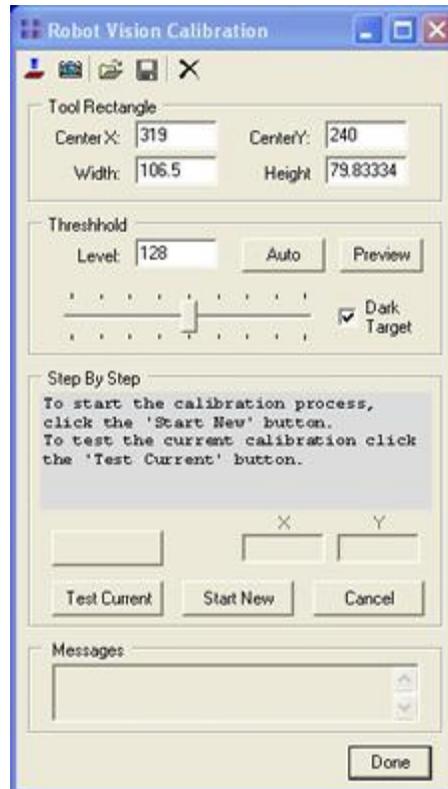
The function and general content of each of the buttons and sub-panels are described in the following
tables.

	Calibration Panel Toolbar	
Icon	**Tool Tip Title**	**Description**
	Live Video	Continuously updates the displayed camera image to show a "live" image.
	Single Picture	Captures a single picture (snap shot) from the selected camera.
	Load Robot Calibration	Displays a pop-up file folder window that permits an existing vision calibration file to be selected and loaded. This allows an old calibration file to be loaded and tested using the "Test current" button described below.
	Save Robot Calibration	Displays a pop-up file folder window that permits the current calibration data to be saved to a disk file.

	Reset Calibration	Provides a way to reset the camera calibration to its default (unity scaling).
✕		

The **Tool Rectangle Sub-Panel** is used for manually positioning and sizing a window around the pattern of squares on the Standard Calibration Target Sheet. When the rectangle is correctly positioned, the system can automatically and accurately determine the position and size of each of the enclosed squares. Normally, this rectangular window is graphically positioned and sized.

Tool Rectangle Sub-Panel	
Field/Button	**Description**
CenterX	Displays and defines the center of the rectangular window.
CenterY	
Width	Displays and defines the size of the rectangular window.
Height	

The **Threshold Sub-Panel** defines the threshold for binary images. Binary images are used in combination with grayscale images for locating the Calibration Target Sheet squares and for other aspects of the calibration process. Binary processing allows distinct high contrast objects to be easily located whereas the grayscale processing produces the accurate final results. The binary threshold level specifies the image brightness value that defines whether a pixel is considered to be black or white.

Threshold Sub-Panel	
Field/Button	**Description**
Level	Displays and sets the binary threshold value. This value can range from 0 to 255.
Auto Button	Automatically adjusts the binary threshold by sampling the pixel values in the area bounded by the Tool Rectangle.
Preview Button	Holding down this button displays the area bounded by the Tool Rectangle in binary mode. This permits you to preview the effect of the current threshold level.
Slider Control	Allows manual adjustment of the binary threshold level.
Dark Target	Inverts the black and white color of each binary pixel. The Standard Calibration Target Sheet has black squares on a white background. For this sheet, the Dark Target box should always be checked. If during any portion of the calibration process a white target is placed on a dark background, this box should be un-checked.

The **Step-by-Step Sub-Panel** displays the instructions for executing each step of the calibration process and contains buttons for moving to the next or the previous step.

Step-by-Step Sub-Panel	
Field/Button	**Description**
Instruction window	Displays the instructions to be performed in the next step of the calibration process.
Test Current	Tests the current calibration data for the selected camera. This process locates the centroid of an object in the Tool Rectangle and translates its location into the robot coordinate system by applying the current calibration data. The robot location is displayed in the

	Message Window.
Start New	Initiates a new calibration process. This starts the Step-by-Step procedure.
Cancel	Aborts the calibration procedure and returns to the introductory panel for the Step-by-Step process.

The **Message Window** and **Main Buttons** display informative and error messages as the calibration process is executed and allows you to exit the process.

Message Window and Main Buttons	
Field/Button	**Description**
Message window	Displays informative and error messages as the calibration process is executed.
Done	Terminates the calibration procedure and closes the pop-up window.

Calibration Data Files

Calibration information is stored in disk files with a ".dat" extension. By default, these files are stored in the "ExampleProjects" folder in the PreciseVision directory. For example, for the default installation directory for PreciseVision version 1.2, the calibration files will be stored in the following directory:

C:\Program Files\Precise Automation\PreciseVision 1.2\ExampleProjects

After a calibration data disk file has been created, it has to be specified in each Vision Project in which it is to be used. When a Vision Project is opened, to associate a calibration file with a camera, open the Camera Calibration Configuration pop-up by selecting **Edit > Camera Calibration Configuration**. The pop-up panel should appear as follows:
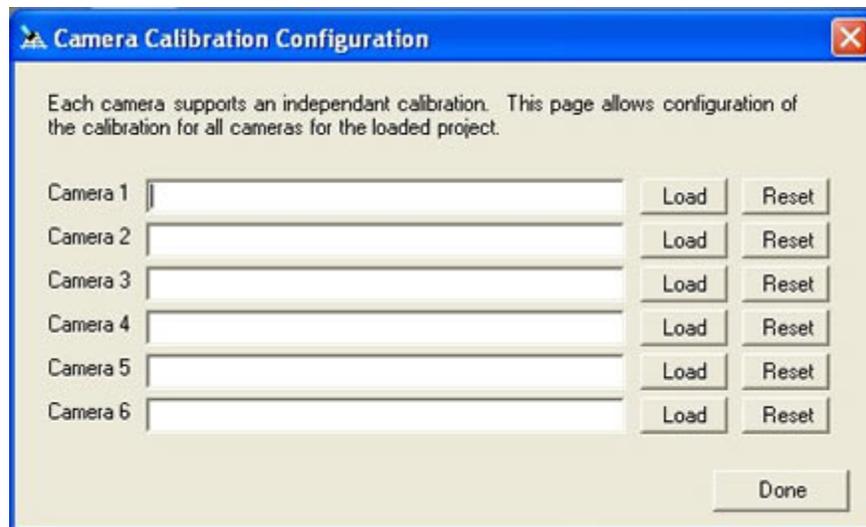
Click the Load button to select a calibration file to be associated with a camera. A dialog window will be displayed that will allow you to browse to the directory location that contains the calibration (.dat) file. If you wish to delete an association between a camera and a calibration file, click on the Reset button.

Once a camera calibration file is specified in the Camera Calibration Configuration menu, the calibration will automatically be applied whenever a picture is acquired from the referenced camera within the Vision Project. Also, whenever the Vision Project is opened, the specified calibration files will be automatically opened and reloaded.

Area Camera Calibration

The Area Camera Calibration method defines the parameters for converting from a camera's pixels coordinate system to units of millimeters. In addition, this procedure adjusts for perspective distortion, which occurs when the camera is not perfectly perpendicular to the plane of the objects to be located. The origin of the both coordinate systems are at the bottom left-hand corner of the camera image. This calibration method is appropriate for testing the vision system, using simple metrology operations or for other applications not needing a robot. However, since this method does not include a transformation between the camera frame of reference and the world coordinate system of a robot, this method is not sufficient for vision-guided robot applications.

This calibration method relies upon the Standard Calibration Target Sheet and does not require a robot or any automated motion.

Prior to executing this calibration method, the following steps must first be performed:

- The "General Calibration Instructions" section of this manual should be read.
- The Standard Calibration Target Sheet must be printed and available.
- The camera must be rigidly mounted, selected, and accurately setup in terms of its focus, aperture setting, and gain and offset.

The following instructions and pictures are an enhanced version of the directions that are displayed in the Step-by-Step informational window of the calibration method pop-up window.

> » To start this calibration method, select *Tool > Calibrate Vision > Area Camera Calibration.*
>
> » Press the *Live Video* icon in the Calibration Popup Panel toolbar to continuously update the camera image.
>
> » Place the Standard Calibration Target Sheet in the same focal plane that you expect the primary part features to be located. Position the sheet such that one of the patterns is fully visible in the field-of-view. If multiple patterns are fully visible, the largest of these should be used during the calibration process.

At this point, the Vision Camera Display and the Calibration Pop-up panel should resemble the following.
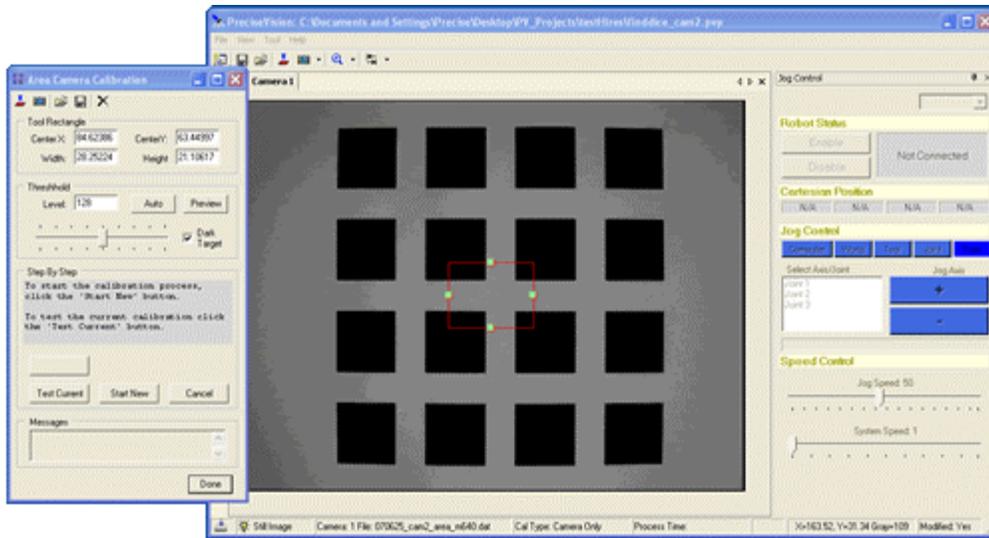
» Press the *Start New* Button to initiate the Step-by-Step process.

» Position and size the Vision *Tool Rectangle* around the perimeter of the calibration target pattern, ensuring that all squares in the pattern are completely within (not touching) the vision tool rectangle as illustrated below.


» Adjust the *Threshold Level* such that the binary image displayed in the Vision Tool Rectangle accurately represents the calibration target pattern. In most cases, the *Auto* button can be pressed to automatically set the threshold. Press the *Preview* button if the binary image is not being displayed.
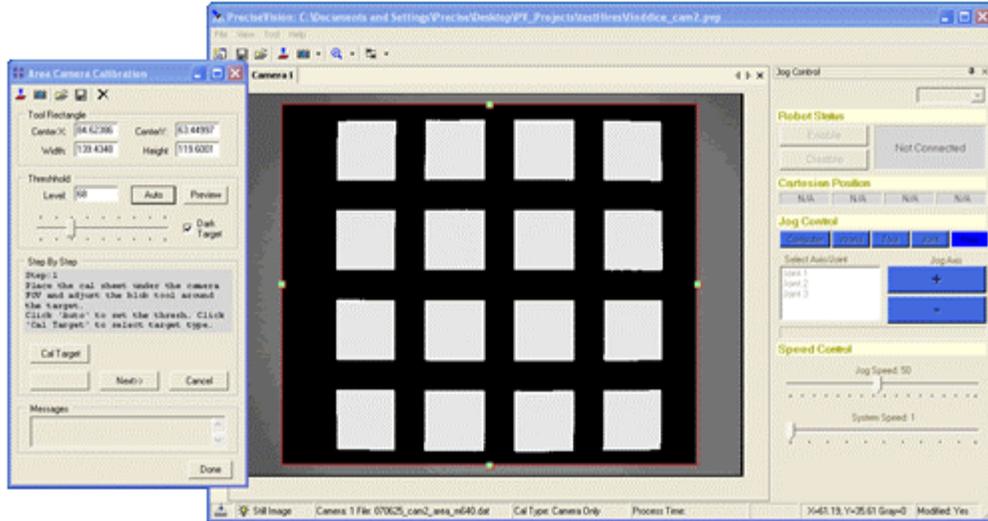
> » If you are not using the largest target pattern, press the *Cal Target* button and select one of the predefined target sizes or enter the correct dimensions and spacing (pitch) for the pattern. The pattern dimensional information can be found in the lower left-hand corner of the Target Sheet.
>
> » Click the *Next* button to go to **Step 2**.
>
> » Press the *Calibrate* button to perform the calibration process.
>
> » Read the contents of the *Message* window to verify that no error has occurred.

The Vision Camera Display window should now resemble the following picture. Each of the four sides of each square should have been automatically and accurately located with a Line Fitter tool. If the Message window indicates that an error has occurred, most likely, either the *Threshold* value or the *Dark Target* check box were not set correctly.
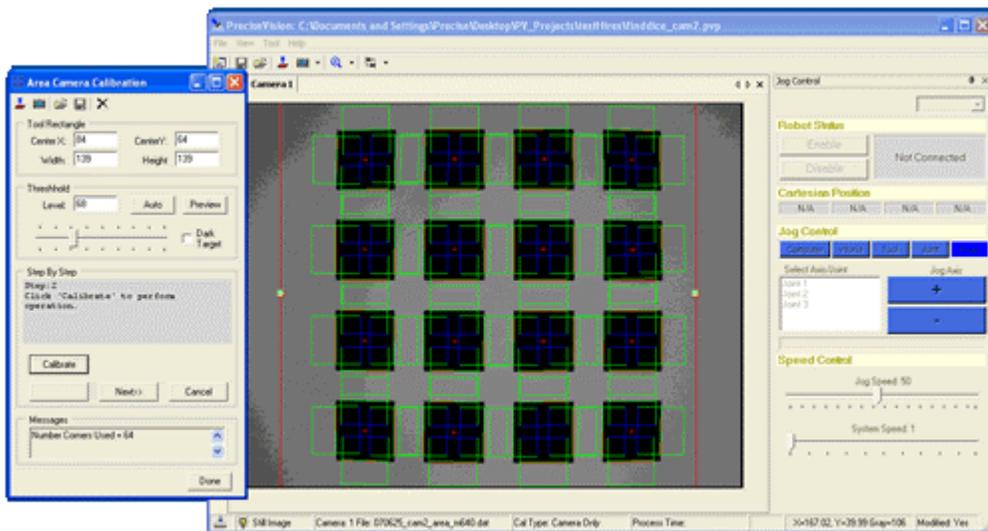
> » Click the *Next* button to go to the final step and save the new calibration data in a ".dat" file.
>
> » Click the *Done* button to terminate the calibration process.
>
> » Finally, you should follow the instructions in the "Calibration Data Files" section of the "General Calibration Instructions" to have this calibration data automatically loaded when your Vision Project is loaded.

This completes the Area Camera Calibration procedure.

Robot Vision Area Camera Calibration

The Robot Vision Area Camera Calibration method defines the parameters for converting from a camera's pixel coordinate system to units of millimeters and also determines the transformation from the camera's frame of reference to the robot's world coordinate system. This calibration method is appropriate for vision guided robot applications where a stationary area camera is utilized to locate and inspect parts.

In this calibration method, two distinct procedures are performed. First, the camera pixel position to millimeter relationship is established using the same Calibration Target procedure as in the Area Camera Calibration method. Second, the camera reference frame to robot world reference frame relationship is determined by manually moving a calibration disk to several positions in the camera's field of view. At each position, you manually move the robot so that an attached pointer touches the center of the disk.

For the first part of this method, no robot motion is required. However, for the second part of this method, the robot's controller must be powered up and the robot must be manually moved to positions within the camera's field of view.

Prior to executing this calibration method, the following steps must first be performed:

- The "General Calibration Instructions" section of this manual should be read.
- The Standard Calibration Target Sheet must be printed and available.
- The camera must be rigidly mounted, selected, and accurately setup in terms of its focus, aperture setting, and gain and offset.
- A circular, relatively flat calibration disk and a pointer mounted to the tool flange of the robot must be obtained. The calibration disk's diameter is recommended to be roughly 10-15% of the width of the camera's field of view. This calibration method relies upon locating the center of the disk at several different positions. Consequently, if the disk is too large, its size would limit the maximum distance between center points. The color of this disk should be selected to maximize its contrast with the background. The disk should also have a feature that aids the user in aligning the center of the pointer with the center of the disk.
- The Guidance Controller for the robot must be powered on and interfaced via Ethernet to the PC that is executing PreciseVision.
- The IP address for the Guidance Controller must be defined by going to **File > Preferences > Client Settings**. This setting allows the calibration procedure to automatically retrieve the robot's location when required. This IP setup procedure is not necessary when executing a vision guidance application.
- The robot must be homed and capable of being manually moved. However, while it is convenient, it is not necessary for the robot to be moved automatically or under power.

> ⚠️ **DANGER**: Before proceeding with this procedure, please ensure that the robot has been properly mounted, all required safety interlocks have been installed and tested, and power has been connected. All safe guards required to permit the robot to be manually or automatically repositioned must be in place. For the PrecisePlace robots, this information is provided in the *"PrecisePlace 2300/2400 Robot, Hardware Introduction and Reference Manual"*.

The following instructions and pictures are an enhanced version of the directions that are displayed in the Step-by-Step informational window of the calibration method pop-up window.

> » To start this calibration method, select *Tool > Calibrate Vision > Robot Vision Area Camera Calibration.*
>
> » Press the *Live Video* icon in the Calibration Popup Panel toolbar to continuously update the camera image.
>
> » Place the Standard Calibration Target Sheet in the same focal plane that you expect the primary part features to be located.   Position the sheet such that one of the patterns is fully visible in the field-of-view. If multiple patterns are fully visible, the largest of these should be used during the calibration process.

At this point, the Vision Camera Display and the Calibration Pop-up panel should resemble the following.
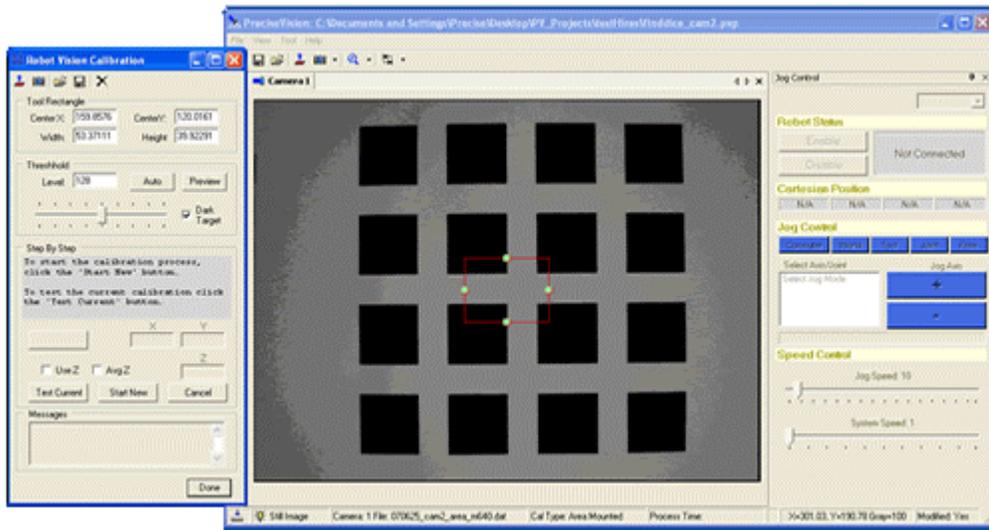
> » Press the *Start New* Button to initiate the Step-by-Step process.

A pop-up will now be displayed that will permit you to establish a network connection to the robot controller. This connection is required if you wish to move the robot using a special compact version of the web Virtual Manual Control Pendant that is built into PreciseVision (right panel in picture below) or to allow PV to automatically move the robot during the calibration process. If you intend to move the robot using the web Virtual Manual Control Pendant or by some other means, this network connection is not required in order to successfully perform the remainder of the calibration procedure.
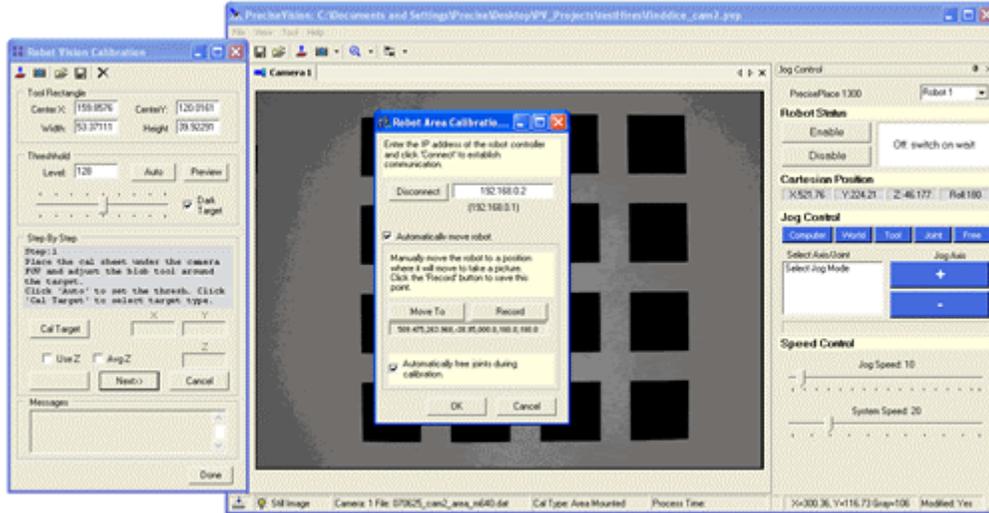
> » Optionally press the **Connect** Button if you wish to move the robot using PV's manual control panel or to permit PV to automatically move the robot during the calibration process.
>
> » Optionally check the **Automatically move robot** box and record a safe location where the robot should be moved whenever a picture is to be taken. This location should position the robot so it does not obstruct the camera's field of view when a picture of the calibration area is to be taken.
>
> » Optionally check the **Automatically free joints during calibration** if you want the robot's joints to be "freed" when the robot must be manually positioned. If selected, this will automatically free all of the joints of the robot when the robot's tool tip must be manually positioned above the calibration disk.

After the pop-up is closed, if you have connected to the controller, the "Jog Control" panel on the right will become active. Once active, this panel permits the robot to be easily repositioned in World, Tool, Joint and Free modes.

> » Position and size the Vision **Tool Rectangle** around the perimeter of the calibration target pattern, ensuring that all squares in the pattern are completely within (not touching) the vision tool rectangle.
>
> » Adjust the **Threshold Level** such that the binary image displayed in the Vision Tool Rectangle accurately represents the calibration target pattern. In most cases, the **Auto** button can be pressed to automatically set the threshold. Press the **Preview** button if the binary image is not being displayed.
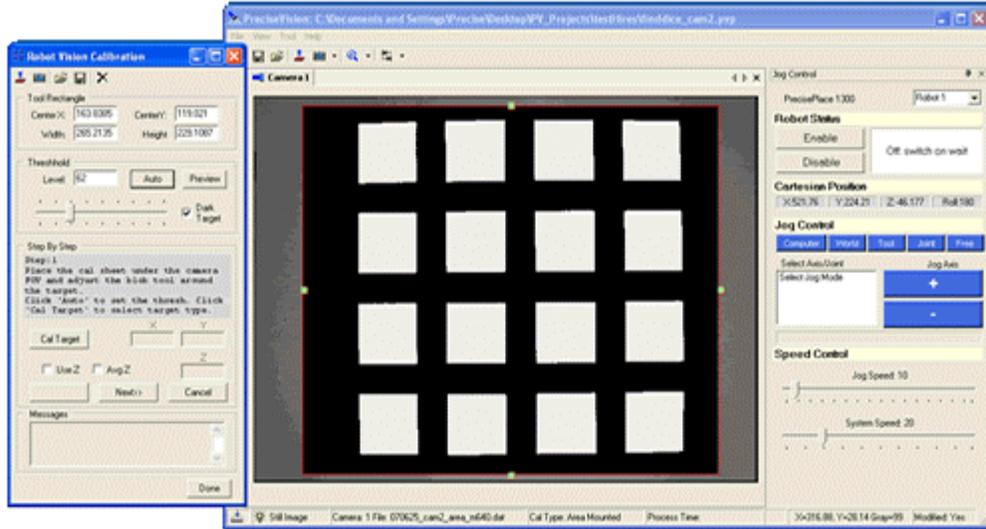
> » If you are not using the largest target pattern, press the *Cal Target* button and select one of the predefined target sizes or enter the correct dimensions and spacing (pitch) for the pattern. The pattern dimensional information can be found in the lower left-hand corner of the Target Sheet.
>
> » Click the *Next* button to go to **Step 2**.
>
> » Press the *Calibrate* button to perform the Target Sheet calibration process.
>
> » Read the contents of the *Message* window to verify that no error has occurred.

The Vision Camera Display Window should resemble the following picture. Each of the four sides of each square should have been automatically and accurately located with a Line Fitter tool. If the Message window indicates that an error has occurred, most likely, either the *Threshold Level* or the *Dark Target* check box was not set correctly.
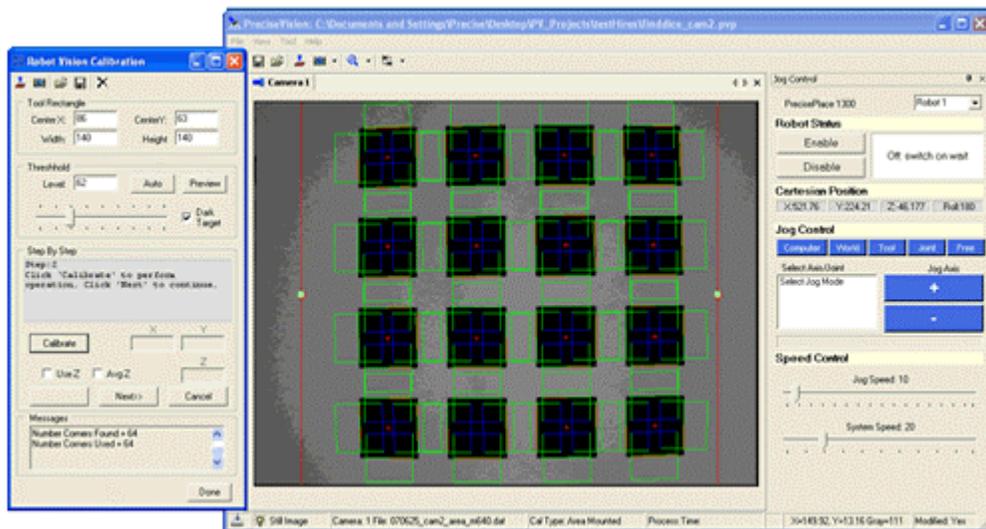
The calculation of the conversion from camera pixel positions to millimeters has been completed.

Next, in the process described below, the calibration disk must be placed into each of the four corners of the camera's field-of-view. When you do so, ensure that the disk is entirely in the camera's field-of-view and is at a position that can be reached by the robot. The closer that the disk is to each corner, the better the results. However, if the disk cannot be placed very close to a corner, the calibration will still be valid, just not quite as accurate.

At each disk position, the robot will have to be moved until its pointer is just touching the center of the disk. For this task, the robot can be moved under servo control via PV's Jog Control or the robot may be positioned manually by disabling high power and pushing the axes into position. If a PrecisePlace robot is being used and high power is disabled, the brake release button in the Z-axis cover must be pressed in order to position the Z-axis.

> ⚠️ **WARNING**: If you have disabled high-power and are in the working envelope of a robot, it is strongly recommended that you leave the E-stop button depressed to ensure that power to the robot cannot be enabled while a person is in contact with the mechanism.

During the calibration process, the system can optionally record the Z height of each of the disk positions and include this information in the calibration data. This permits the vision system to return the correct Z positions for parts even if they are on an inclined surface. To select this option, the *"Use Z"* box must be checked. Alternatively, if the part surface is in the X-Y plane and you wish to include the Z height information in the calibration but ignore sampling deviations, the *"Use Z"* and the *"Avg Z"* boxes should both be checked.

To begin the calibration sequence with the calibration disk, perform the following operations.

> » Click the *Next* button to go to **Step 3**.
>
> » Typically, the *"Use Z"* box should be checked. In addition, the *"Avg Z"* box should be checked if the field-of-view is in the X-Y plane of the robot's world coordinate system and you wish to average out small Z height sampling errors.
>
> » Place the calibration disk in one corner of the camera's field-of-view and in a position that can be reached by the robot.

The Calibration Panel and the Camera Display Window should resemble the picture shown below. You will note that the *Tool Rectangle* is displaying its contents in binary mode. This mode is used to initially locate the calibration disk. As a convenience, if the camera's field-of-view is clear of other parts, the *Tool Rectangle* can be set large to enclose the disk in all four of its anticipated positions and can be left in this position and size for the remainder of the calibration procedure. However, if there are visual obstacles that may confuse the vision system when it attempts to locate the disk, the *Tool Rectangle* should be reduced to a size just larger than the disk and should be repositioned each time the disk is moved to fully enclose the disk.
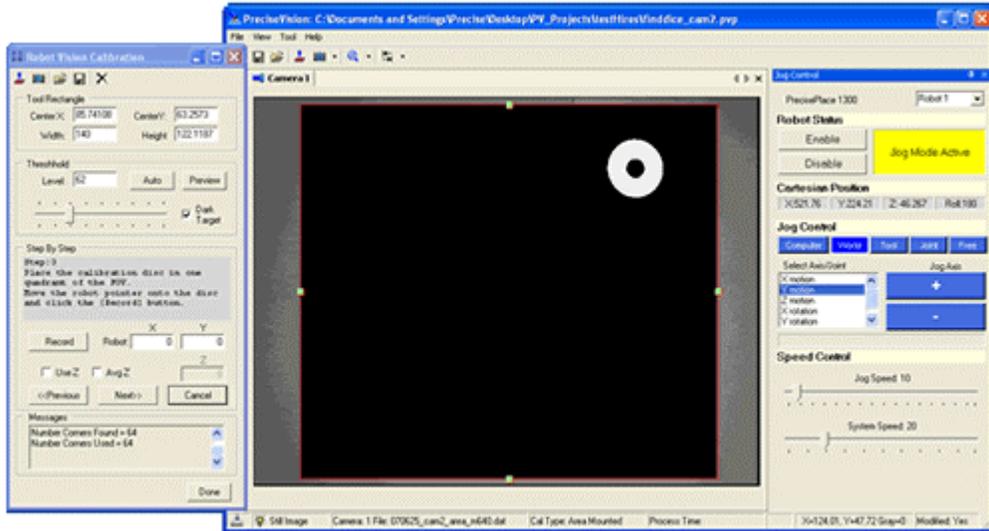
» Manually move the robot pointer until it is just touching the center of the disk.

» Click the *Record* button in the Step-by-Step panel to have PreciseVision read and record the robot position.

» Click the *Next* button to go to **Step 4** .

» Carefully move the robot off of the calibration disk and avoid moving the disk. Stop the robot anywhere that it is clear of the camera's field-of-view. If the disk moves, go back to **Step 3**. If you have chosen to automatically move the robot back to the original starting point, a prompt will be displayed for this operation. Click "Yes" and the robot will retract and move out of the way automatically.

» Ensure that the *Tool Rectangle* fully encloses the disk and that the *Threshold Level* is properly set to accurately depict the disk in binary mode.

» Press the *Locate* button to take a picture and visually capture the position of the disk.

If the disk is properly located, PreciseVision will draw a smaller circle within the outline of the calibration disk as shown below. If PreciseVision draws a circle anywhere else in the image or does not draw a circle on the disk, an error has occurred. Most likely, the *Threshold Level* was set incorrectly or the *Dark Target* checkbox is set to the wrong sense or some other object in the *Tool Rectangle* confused the system. If this occurs, correct the problem and press the *Locate* button again.

PreciseVision
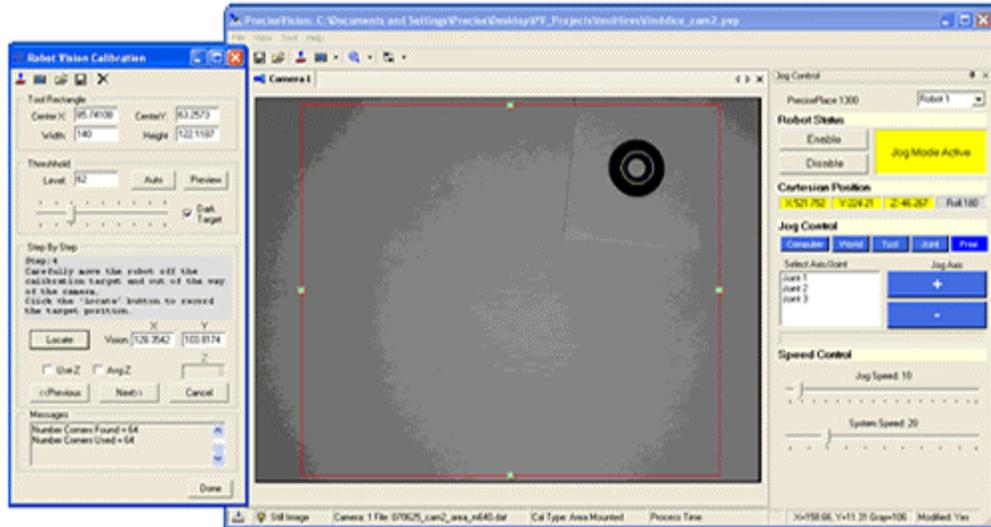
> » Move the disk to the other 3 corners of the camera's field-of-view and repeat the process of recording the robot's position and the disk's position by executing the Step-by-Step procedure's **Step 5** though **Step 10**.

After pressing the *Locate* button on **Step 10**, the Calibration Panel and the Camera Display Window should resemble the following picture that indicates that all four disk positions have been properly located. Note that four smaller circles have been drawn into the four corners of the **Tool Rectangle**.
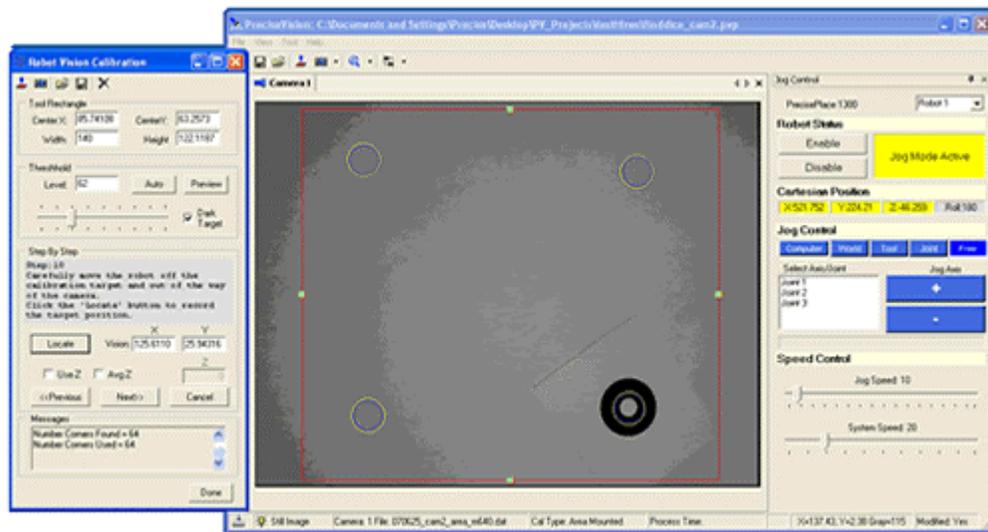
> » Click the *Next* button to go to **Step 11** .
>
> » Click the *Finish* button in have the system compute the final camera calibration data.
>
> » When the file pop-up is displayed, save the new calibration data in a ".dat" file.
>
> » Click the *Done* button to terminate the calibration process.

146

> **»** Finally, you should follow the instructions in the "Calibration Data Files" section of the "General Calibration Instructions" to have this calibration data automatically loaded when your Vision Project is loaded.

This completes the Robot Vision Area Camera Calibration procedure.

If you wish to validate the derived calibration data, you can move the calibration disk around the camera's field of view and press the **Test Current** button. This will display the centroid of the disk in the X and Y boxes in the robot's world coordinate system. These values can be compared to the robot's location when the pointer is positioned at the calibration disk.

Appendix A: FAQ

Frequently Asked Questions

This section contains a compilation of frequently asked questions related to PreciseVision.

A. General Questions

1. [What do you do if PV was working but will no longer properly start up?](#)
2. [How do I transfer my PV license to another host computer?](#)

B. DALSA Cameras

1. [What does the DALSA camera LED indicate?](#)
2. [What do you do if you are not able to get an image with a DALSA camera?](#)
3. [What do you do if other NIC's stop operating after DALSA drivers are installed?](#)
4. [How do you eliminate diagonal stripes in the DALSA camera image?](#)

General Questions

What do you do if PV was working but will no longer properly start up?

Sometimes the preferences information for PreciseVision can become corrupted or can reference a PV project that is corrupted.   When this occurs, PV will fail to launch. To work around this problem, do the following.

1.
2. Locate the folder where PreciseVision is installed. Typically, this will be "C:\Program Files\Precise Automation\PreciseVision V__\".
3. Rename your preferences file "Preferences.vpr" to "Preferences.bad".
4. Restart PreciseVision.

If PreciseVision starts properly, please email your bad Preferences file and your PreciseVision projects to support@preciseautomation.com with a short explanation so we can diagnosis the problem.

How do I transfer my PV license to another host computer?

The PreciseVision software is only licensed to execute on a single PC. In general, if you need to execute PV on additional PC's, additional licenses must be purchased.

However, if the PC on which PV is licensed to execute is being obsoleted, you can transfer a purchased PV license to a new PC.

Note, as soon as you execute the following procedure, PV will no longer be permitted to execute on your old PC.

To transfer a purchased PV license to another PC, perform the following procedure.

1.
2. Follow the standard procedure for installing PV on the **new** PC.
3. On the **new** PC, execute the procedure for licensing PV to obtain the PC's ID information. This is needed to issue you a new license for this PC.
4. Execute PV 2.0 or later on the **old** PC. If you are executing an older version of PV, obtain a new version from the Precise Support website and install it on the **old** PC.
5. Select *"Help > Product Activation"*.
6. Click the *"Uninstall"* button.
7. Follow the instruction to execute the uninstall procedure. *After you execute this procedure, you will no longer be able to execute PV on the **old** PC.*
8. Email the following information to sales@preciseautomation.com and request that the licence be transferred.
    9.
    o
    o   Your company name
    o   The name of the individual that the license was issued to, if you know
    o   The original license code displayed by the uninstall procedure
    o   The uninstall code displayed by the uninstall procedure
    o   The new PC's ID displayed by the licensing procedure
10. Once you obtain your new license key, execute the standard activation process on the **new** PC.

This completes the license transfer process.

DALSA Cameras

What does the DALSA camera LED indicate?

The back surface of the DALSA camera that contains the Ethernet and DC power plugs includes an LED that indicates the current status of the camera. The interpretation of the LED's color and blinking state is described in the following table.

LED State	Camera Status
Off	Camera is not receiving power
Steady Red	Camera is not initialized
Slow Flashing Red	Camera initialization problem. Most likely indicates a camera hardware failure.
Fast Flashing Red	Camera is too hot
Slow Flashing Blue	Waiting for an IP address
Fast Flashing Blue	Ethernet cable disconnected (no link)
Steady Blue	IP address assigned, PreciseVision not connected to camera

| Steady Green	PreciseVision connected to camera, standard operational state
Slow Flashing Green	Camera waiting for hardware trigger to take picture
Fast Flashing Green	Continuous acquisitions in progress, standard live video state

What do you do if you are not able to get an image with a DALSA camera?

If you are not able to get a image from a DALSA camera from within PreciseVision, there are several possible sources for this problem. Most of the problems and remedies described below apply to systems that are being executed for the first time, but some are appropriate for systems that have already operated.

A. The DALSA camera may not be configured as the hardware to use within PreciseVision.
PreciseVision can operate with a variety of Ethernet camera devices and can even operate in a demonstration mode without a camera. However, at any given time, only one camera type is permitted. The following procedure verifies that PreciseVision is configured to capture images from DALSA cameras.

1. Launch PreciseVision, typically by selecting *Start > Programs > Precise Automation > PV_ > PreciseVision.*
2. Open the Preferences pop-up by selecting the following within PV: *File > Preferences.*
3. Select the "Camera Hardware" tab in the Preferences pop-up window.
4. Ensure that the "Genie" option is selected. You can also click on "Genie Mapping" to specify the logical camera number associated with each physical camera.
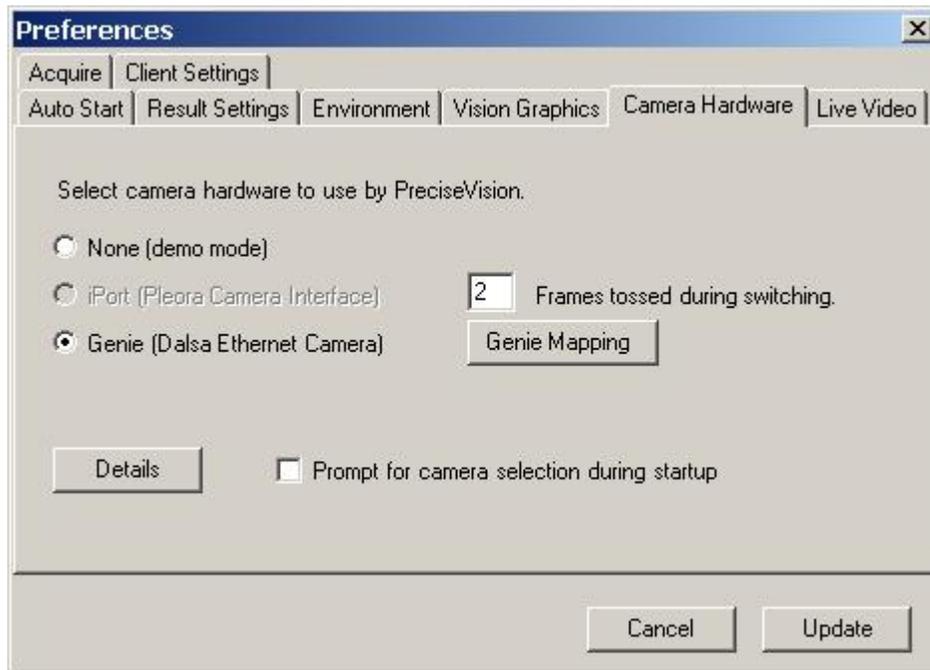
B. <u>**The required software drivers may not be installed.**</u> The use of the DALSA cameras with PV requires the installation of three separate software packages on a Windows PC system.

1. To verify that all three software packages have been installed, bring up the Windows Control Panel typically by selecting *Start > Settings > Control Panel.*
2. Select *"Add or Remove Programs".*
3. Verify that the following packages are installed: (i) DALSA Coreco Sapera LT, (ii) DALSA Genie Framework, (iii) DALSA Sapera Network Imaging Package, and (iv) PreciseVision.
4. If any of these packages are missing, refer to the PV installation instructions for installing the missing application packages.

C. <u>**No cameras found by the PC**</u>. Typically after the software is installed and the PC is rebooted, any DALSA cameras will immediately be located by the PC and be accessible. However, if you click on the small camera icon that is displayed in the Window's start bar and see the <u>"No Device Found"</u> message in the Device's pop-up, it means that the PC could not locate a camera on its local Ethernet Network.


1. Verify that the camera is receiving power and is properly connected to the network. The LED on the camera should have initially been red and then changed to a blue color. If the LED is not lit or is red, either the camera is not receiving power or is defective or is not connected to the network.
2. If the camera LED is blue, the PC's firewall or virus protection is probably blocking the camera connection. Disable your PC's firewall and turn off any virus protection. If the camera becomes visible by the PC, go to the DALSA website (www.imaging.com) for information on how you still might be able to enable your virus protection or firewall.

D. <u>**The camera may not have an IP address compatible with your PC**</u>. Normally, the camera will automatically obtain an IP address from a DHCP server within your network. However, a DHCP server may not be available or the camera may have been set to a fixed (Persistent) IP address that is not consistent with your network. If you click on the small camera icon that is displayed in the Window's start bar and see <u>"IP Error"</u> in the Status area of the Device's pop-up, it indicates that the IP address of the camera is not set compatibly with your Ethernet network.

1. Execute the "*DALSA Network Configuration Tool*". Typically this is launched by selecting *Start > Programs > DALSA > Sapera Network Imaging Package > DALSA Network Configuration Tool.*
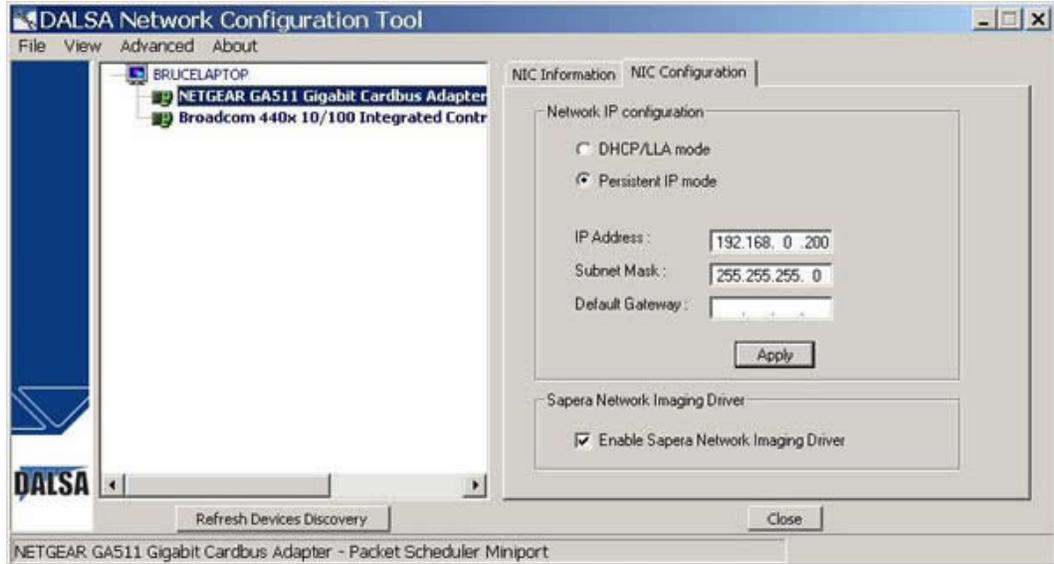
2. Click on the Ethernet network interface card (NIC) that is connected to the camera.
3. Select the "NIC Configuration" tab on the right. This tab will permit you to change the IP address of your PC. Change the IP address and subnet mask to be compatible with the address of the camera as displayed in the Devices window shown above. For example, in the previous picture, the camera IP address was 169.254.241.145. IP addresses whose first number are between 128-191 fall into Class B and require a subnet mask of 255.255.0.0. Set a compatible IP address and subnet mask and click "Apply".
4. After you have selected a compatible address, the camera information should be displayed in the left hand window below the NIC. If you click on the camera information, its IP information will be displayed on the "Device Configuration" tab as shown below.
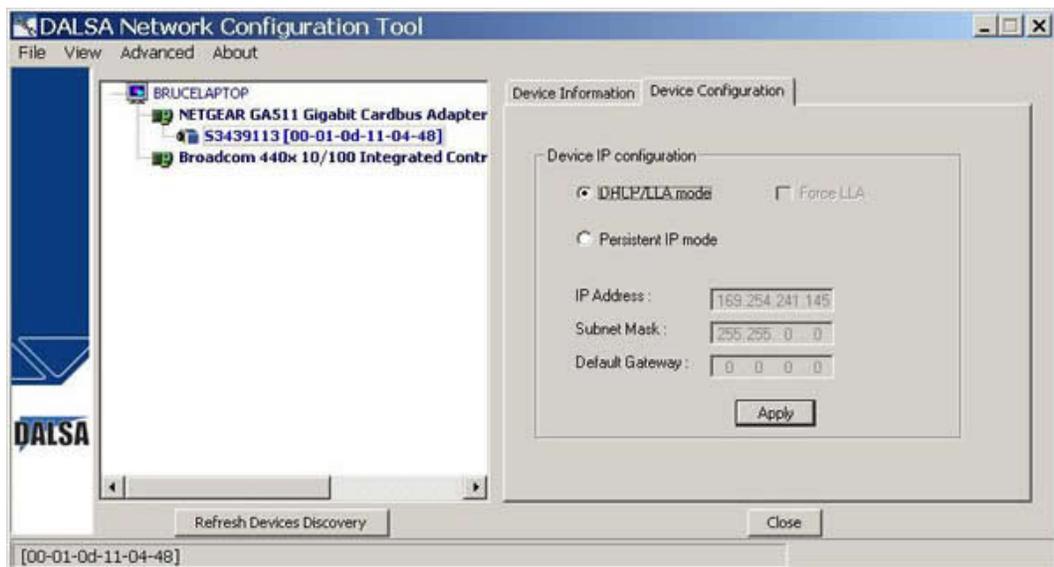
5.   Enter the desired IP address and subnet mask for the camera and press "Apply".
6.   Repeat steps 2 and 3 to reset your PC to the previous IP information.
7.   Close the Devices window and double click on the camera icon in the start bar to redisplay this window. The status should no longer indicate an IP Error.

E.   The camera Ethernet drivers may be incorrectly configured. If the camera and its Windows drivers are compatible and properly configured, the CamExpert Windows applications should always be able to display a live camera image.

1.   To test the camera and its Windows driver, launch the DALSA CamExpert, typically by selecting *Start > Programs > DALSA > Sapera LT > CamExpert.* You should see the following application window.
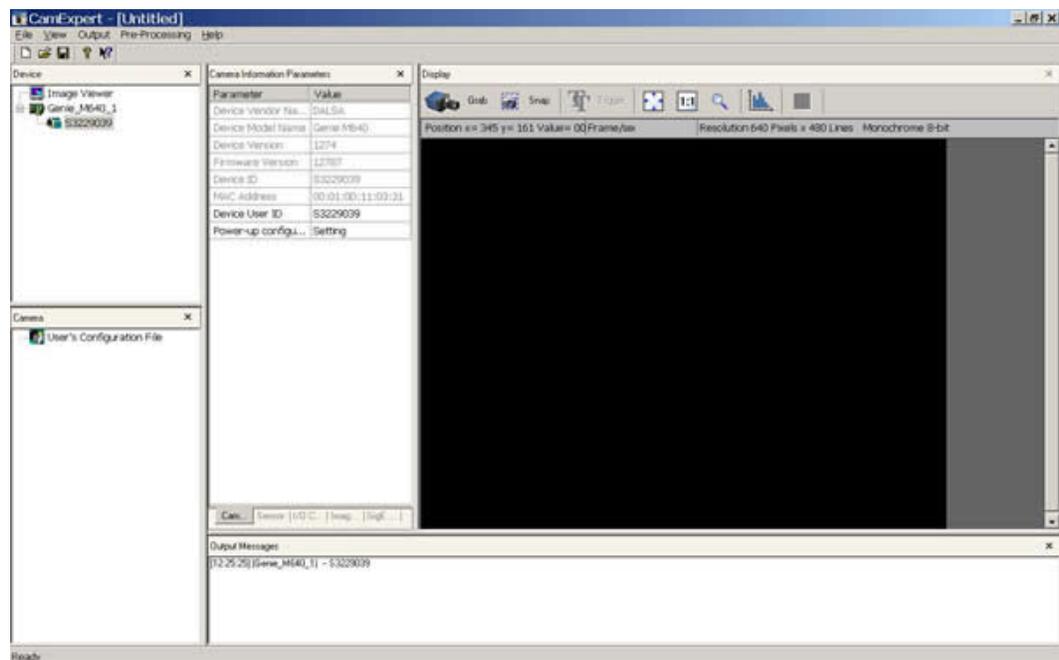
2.   Click on the camera listed in the top-left hand window and click the "*Grab*" button above the camera display window.
3.   If the camera and its Windows drivers are operating properly, you should see live camera video in the camera window and the "Frame/sec" above the video should be displaying the frame capture rate. If this is all correct, close the CamExpert and restart PreciseVision **(NOTE: The CamExpert and PV can not be executed simultaneously).** PreciseVision should now be able to capture images.
4.   If the live video output of the camera was not displayed, try disabling unnecessary NIC's (especially wireless NIC's) by opening *My Network Places > View Network Connections*. Right click on unnecessary NIC's and select *"Disable"*.
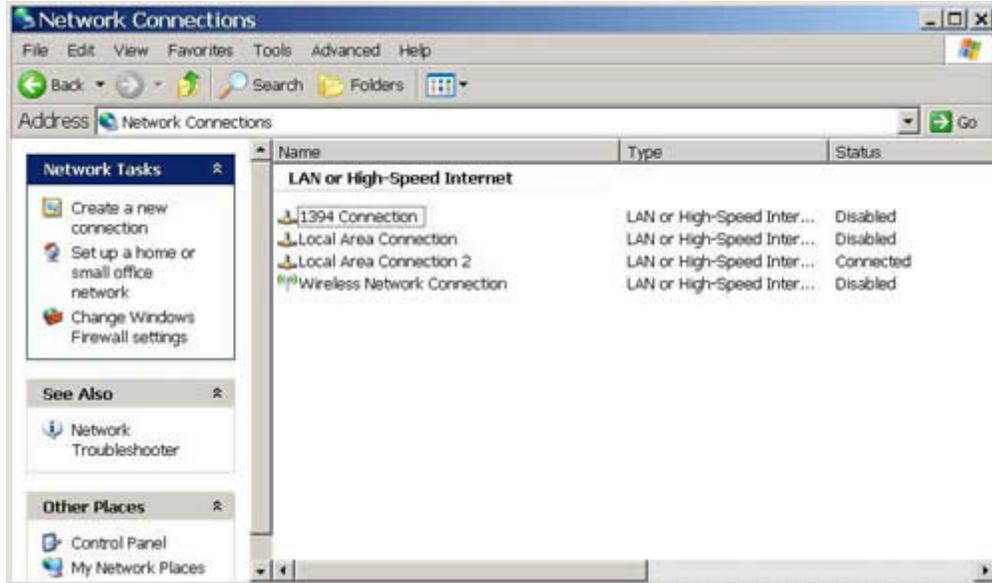
5. If this did not resolve the problem, try disabling all unnecessary software options associated with the NIC connected to the camera. To do this open *My Network Places > View Network Connections*.
6. Double click the NIC attached to the camera to display the "Local Area Connection" window.
7. Click on the "Properties" button.
8. In the Properties window, uncheck all software services except for "CorSnid Driver" and "Internet Protocol (TCP/IP)", and press "Ok".
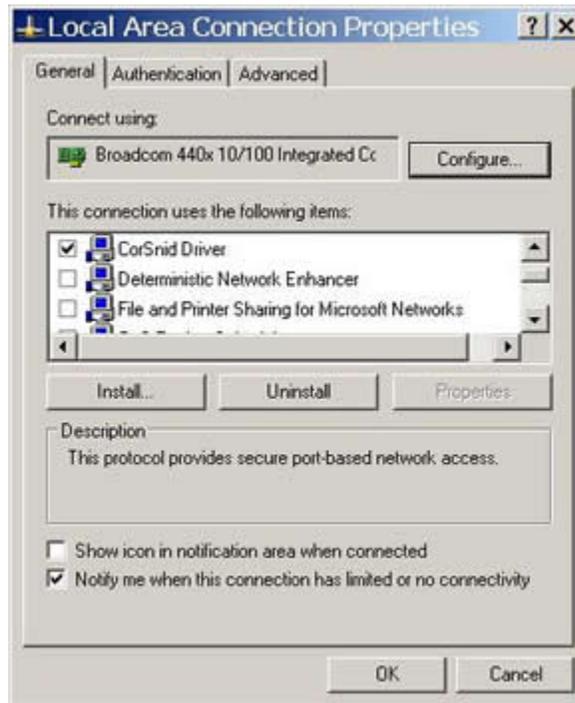
9. If this does not resolve the problem, there may be a fundamental incompatibility between your NIC and the DALSA camera. Please contact Precise Automation for further assistance.

What do you do if other NIC's stop operating after DALSA drivers are installed?

When you install the DALSA camera drivers, they automatically add camera specific software that is executed as part of the driver for every Ethernet card (NIC) in your PC. For certain NIC's, this additional software can cause the Ethernet interface to malfunction even if you do not connect to a camera using this device. This seems to often be the case with wireless Ethernet interfaces in particular.

To disable the DALSA specific software for a NIC, do the following:

1. Access the software options for the NIC by opening *My Network Places > View Network Connections*.
2. Double click the NIC that is causing the problem to display the "Local Area Connection" window.
3. Click on the "Properties" button.
4. In the Properties window, uncheck the software labeled "CorSnid Driver" and press "Ok".
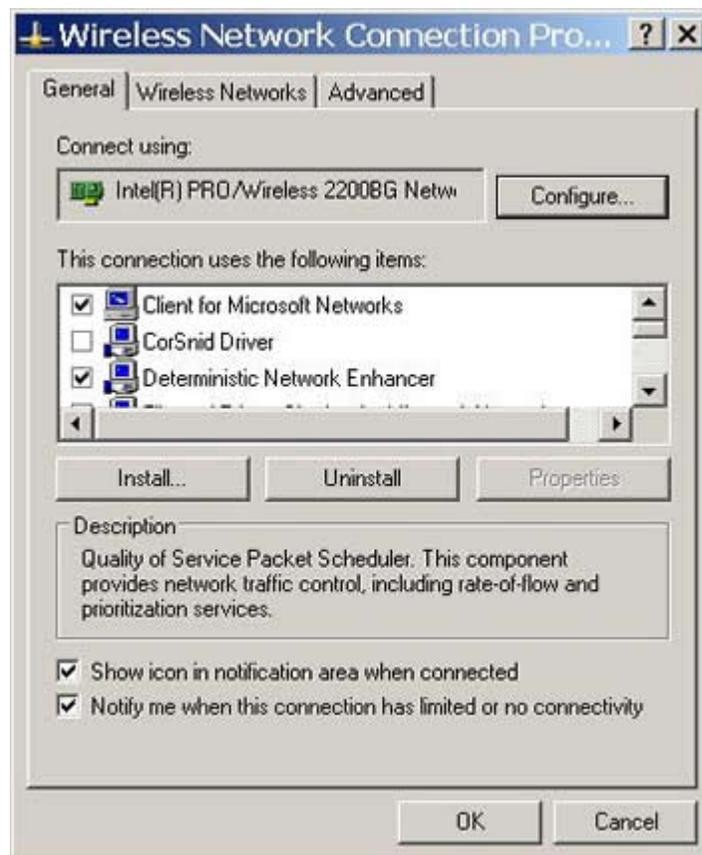
How do you eliminate diagonal stripes in the DALSA camera image?

You may sometimes see a blinking pattern of diagonal stripes in the live video image generated by a DALSA camera. For example, in the image below, the true image should be a grid of horizontal and vertical lines. Interspersed in the grid is a pattern of diagonal lines that is not in the real image.
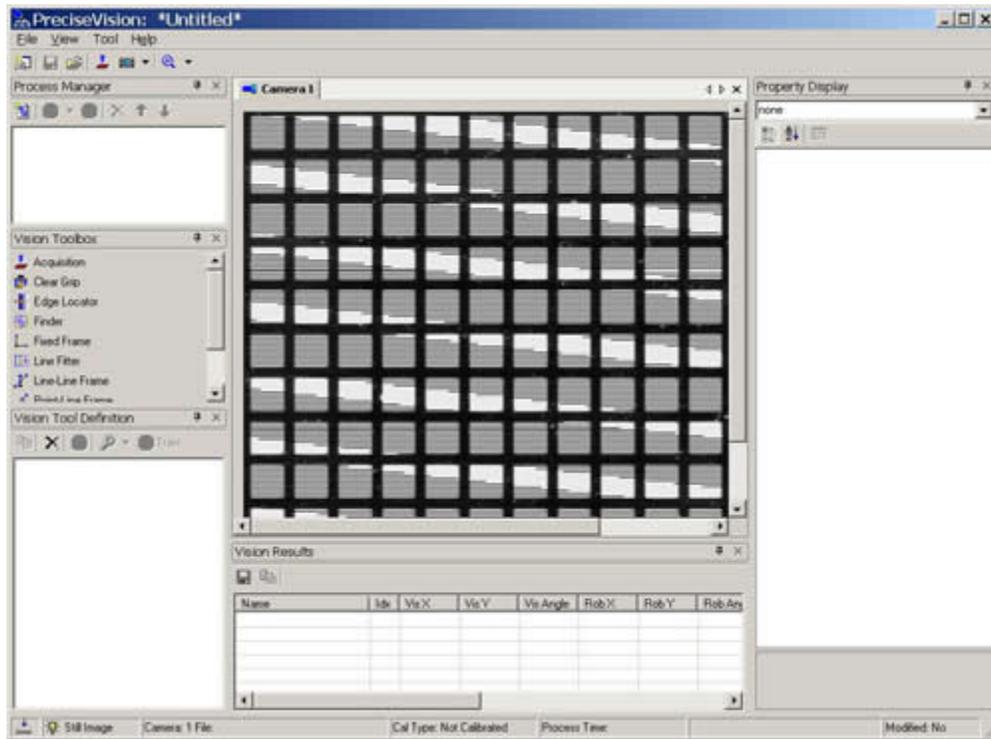
This pattern of lines is due to a problem with the DALSA communications buffer size. To correct this error, do the following:.

1. Exit from PreciseVision.
2. Start the DALSA CamExpert application. Normally, this can be launched from the Start bar by selecting: *Start > Programs > DALSA > Sapera LT > CamExpert*.
3. Press the "Grab" button to verify that the problem can still be seen.
4. Press the "Grab" button again to stop the live image capture mode.
5. In the "Camera Information Parameters" table, select the "GigE Vision" tab.
6. Change the "Network Configuration Mode" from "Automatic" to "Manual".
7. Decrease the size of the "Packet Size (in Bytes)". This value might be set to 1984. Set it to 1500 instead.
8. Press the "Grab" button to verify that the problem has been corrected.
9. Exit the CamExpert application, saving your changes and restart PreciseVision.

Appendix B: Additional Topics

PreciseVision Remote PC Control

Even though PreciseVision executes as a standalone application within a PC environment, its user interface can be integrated with the user interface of another PC application. PreciseVision provides the ability for any PC application to send commands to PreciseVision using TCP/IP. These commands can specify the format of the PV window and its size and position on the screen. This set of commands also includes a few methods for reading and writing data within PreciseVision and for triggering the execution of a vision process. These commands allow any PC application environment to manipulate PreciseVision to look and feel like it is part of the environment without the need for complicated installation of COM objects or distribution issues.

The communication interface uses simple string messages that are passed through an open TCP socket connection. When PreciseVision begins execution, it automatically runs a remote TCP server in a background thread. This thread listens on port (1450) and is dedicated to servicing commands from external applications. This interface can be tested and debugged using any Telnet session connected via the communication port.

Whenever the communication connection is lost, the PV user interface is automatically restored to its normally stand alone mode, i.e. the form title, status, menu and toolbars are all displayed.

Command Definition

Each transmitted command must conform to the following string format:

command [key] arg1 arg2 arg3 … arg20 {LF}

While in Telnet mode, the command response may be as simple as a {LF} or it may be a string that returns requested information. Please see the command description table below for details.

In general, the format of the response returned for each command is formatted as follows:

errorcode arg1 ... argN {LF}

If no error occurs, the returned "errorcode" will have a value of "0" to indicate success. If an error occurs, a negative error code is returned and "arg1" contains the string representation of the error. For some commands, "arg1 ... argN" contains information returned by PreciseVision.

Supported Commands

The following table describes the legal commands. Please note that the **CMD** name is case sensitive.

CMD	KEY	Argument List	Description & Responses

getnumresults		*<tool>*	Retrieves the number of results that exist for the specified tool. The *<tool>* must be of the form *processname.toolname.*
			
			The information returned by this command is as follows:
			
			   <err> <num_results>
			where
			   <err>: Standard error code or 0 for success.
			   <num_results>: The number of results available for this tool.

getresults		*<tool> <index>*	Retrieves the results for a specified tool that was executed as part of a specified process. The *<tool>* must be of the form *processname.toolname.*
			
			The information returned by this command is as follows:
			
			   <err> <x> <y> <z> <yaw> <pitch> <roll> <objectID> <passed> <value> <info_array>
			where
			   <err>: Standard error code or 0 for success.
			   <x> <y> <z> <yaw> <pitch> <roll>: Cartesian coordinates for the located object.
			   <objectID>: Optional index that some tools return to identify the specific instance of an object that is located.
			   <passed>: 0 if tool's inspection was successful or -1 if the inspection failed.
			   <value>: Value of the tool property that was tested by the vision inspection process.
			   <info_array>: Please see the information on the specified tool for a description of any additional result values that are returned.

load	layout	*<file_name>*	Loads a window layout given a specified path and file name. *<file_name>*: The full path and name of a compatible PV layout file. This type of file is created when a layout is manually saved.
load	project	*<file_name>*	Loads a PreciseVision project file from the disk. *<file_name>*: The full path and name of the PreciseVision project file to be loaded. This allows a PC based application to change the project or override the default project loaded when PV starts.
property	set/get	*<property>* *<value>*	Reads or writes the value of a specified property of a specified tool. The *<property>* must be of the form *toolname.propertyname.*

			For write operations, *<value>* defines the new property value that is set. Not all properties are available for reading and/or writing. Please see the vision tool definitions for details on each tool type's properties.
run		*<process>*	Initiates execution of the specified vision process.

set	displaymode	*<mode>*	Dictates the contents and visibility of the PV window. The permitted values for *<mode>* are as follows.
			
			  0: Resets display mode to normal display. PV returns to stand alone mode with all normal windows, toolbars, menus etc. available for use.
			  1: Displays only the camera window and the current Tool's window. The PV form border, title, and status bars are all hidden.
			  2: Displays only the camera window. The PV form border, title, status bars and all dockable controls are all hidden.
			  3: Minimizes the PV window.

set	layout	*<layout_type>*	Selects one of the predefined panel layouts to display within the PV window based upon the value of *<layout_type>*:
			
			  0: Edit mode
			  1: Runtime mode
			  2: Calibration mode

set	position	*<xpos> <ypos> <width> <height>*	Positions and sizes the PV window relative to the screen.
			
			  <xpos>: X position relative to top left of screen
			  <ypos>: Y position relative to top left of the screen
			  <width>: width to set window
			  <height>: height to set window
			
			All units are in pixels.
			
			Note: *<width>* and *<height>* are optional. If not specified, only the X,Y position of the window will be altered.

set	topmost	*<mode>*	Specifies whether the PV window is on top or other windows. The value of *<mode>* is interpreted as follows:
			
			  0: Normal (resets topmost property)
			  1: Keeps window on top.
			
			This allows PV to stay on top of another

			application while the user clicks or drags windows.

set	zoom	*<scale_factor>*	Selects the Zoom scale factor for the camera window. The *<scale_factor>* is interpreted as follows:
			
			  -1: Reset zoom, i.e. no zoom.
			  0-10: Zoom selections from ¼ - 4x