

ARTISAN®
TECHNOLOGY GROUP

Your **definitive** source for quality pre-owned equipment.

Artisan Technology Group
(217) 352-9330 | sales@artisantg.com | artisantg.com

Full-service, independent repair center
with experienced engineers and technicians on staff.

We buy your excess, underutilized, and idle equipment
along with credit for buybacks and trade-ins.

Custom engineering
so your equipment works exactly as you specify.

- Critical and expedited services
- In stock / Ready-to-ship
- Leasing / Rentals / Demos
- ITAR-certified secure asset solutions

Expert team | Trust guarantee | 100% satisfaction

All trademarks, brand names, and brands appearing herein are the property of their respective owners.

Visit our website - *Click HERE*

Installation and User Guide
For
DVT® Vision Sensors



COGNEX

©2001-2006 by Cognex® Corporation.

2nd Edition, First Printing (May 2006). All rights reserved.

Information in this document is subject to change without notice. Companies, names, and data used in examples herein are fictitious unless otherwise noted. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the written permission of Cognex Corporation.

DVT®, SmartLink™, EdgeCount™, FeatureCount™, and ObjectLocate ™ are trademarks of Cognex Corporation.

Microsoft Excel®, Windows®, Windows 2000®, Windows ME®, Windows XP® and Windows NT® are registered trademarks or trademarks of the Microsoft Corporation in the United States and other countries.

Other product and company names mentioned herein may be the trademarks of their respective owners.

Part Ordering Number: DOC-INTL

Document Control Number: MAN-103 REV D

Intellect Release 1.3

Table of Contents

Table of Contents *3*

Chapter 1 - Getting Started *9*

What is Machine Vision? **10**

Imaging and Pixels 10

What is a DVT Vision Sensor? **10**

DVT Vision Sensors Product Family 10

Installing Intellect Software **16**

Connecting to the DVT Vision Sensor **16**

DVT Vision Sensor Hardware Setup **21**

Unpacking and Powering the DVT Vision Sensor 21

Chapter 2 - A Closer Look at the System *27*

System Parameters **29**

Inspection Mode 29

Password Protection 29

Digital I/O Configuration 30

Other Communications Settings 32

Background Scripts 32

Power-on Product 32

Trigger Source 33

FOV Balance 33

Sensor Gain 33

Reflectance Calibration 34

Spectrum Calibration 34

Product Parameters **34**

Exposure Time 34

Digitizing Time 35

Partial Image Window 35

Antiblooming 36

Illumination 36

Product Identification 37

Product Selection 37

System Graphs 38

Product Graphs 40

Vision Tool Parameters **41**

Vision Tool Shape 41
PreProcessing Operations 42
Threshold 43
Vision Tool Result 44
Vision Tool Graphs 45
Digital Relearn 47
Intellect Software 48
Intellect User Interface 48
DVT Vision Sensor Hardware Emulator 52
Chapter 3 - Vision Tool Reference 55
Intellect Vision Tools 56
Counting Vision Tools for Presence/Absence 57
EdgeCounting: Count along a Line Vision Tools 57
FeatureCounting: Count along a Line Vision Tools 58
PatternCounting: Count along a Line Vision Tool 59
Pixel Counting Vision Tools: Intensity and Color Counting 59
Blob Counting: Count in Area Vision Tools: 60
Object Counting: Count in Area Vision Tools: 63
Pattern Counting: Count in Area Vision Tools: 64
Positioning Vision Tools 64
Position along a line/arc 65
Rotation with Vision Tools: Arc and Line Fit 66
Area Positioning Vision Tools 67
Vision Tools used for Specific Inspections 68
Preprocessing Vision Tool 69
Measurement Vision Tools 70
Math Operations: Points and Lines, Statistics Vision Tools 74
Defect Detection Vision Tool 75
Identification Vision Tools 76
Readers 76
Intro to Color Vision Tools: The RGB Color Space 83
Pixel Counting Vision Tool (Color Vision Tool) 83
Color Monitoring 86
Foreground Scripts (Script Vision Tools) 87
Spectrograph Vision Tool 88
Chapter 4 – Inspection Application Guidelines 91

Dealing with Colors **92**
Dealing with Part Movement **93**
Presence/Absence Inspections **95**
Flaw Detection Inspections **96**
Counting Inspections **96**
Measurement Inspections **96**
Techniques for Better SubPixel Measurements 97
Positioning Robotics Applications **97**
Setting up a Coordinate Transformation 98
Color Sorting/Identification Applications **98**
Part Identification **98**
Lot Code and Data Code Applications **99**
OCR 99
1D Reader 99
2D Reader 99
Chapter 5 – DVT Vision Sensor Integration ***101***
Transferring Data from a DVT Vision Sensor **102**
DataLink 102
Modbus Transfers 103
Ethernet Terminal Controller (ETC) 104
Industrial Protocols 105
VDX Driver 105
Appendix A – Emulator tutorial ***107***
Appendix B - Basic TCP/IP Setup ***109***
Appendix C - Advanced TCP/IP ***113***
TCP/IP Installation Tips **113**
TCP/IP Troubleshooting Tips **114**
TCP/IP Subnetting **115**
Appendix D - Upgrading or Reinstalling DVT Vision Sensor Firmware ***117***
To back up a DVT Vision Sensor system to a PC: **117**
To install Intellect on a PC: **117**
Upgrading Intellect firmware on a DVT Vision Sensor **117**
Erasing flash using Telnet **120**
Appendix E – Breakout Boards ***121***
Isolation Breakout Board **121**

Index *123*

Reference Materials for DVT Products

The Installation and User Guide for DVT Vision Sensors is your basic resource for both our hardware and software platforms. This document is designed to provide the users with basic ideas about the functionality of the software.

Several other reference materials available are the:

- DVT Vision Sensor Script Reference Guide, a manual for the programming language used in all scripts in Intellect.
- Intellect HTML Help. These HTML- based help files, which will be installed on your PC's hard disk as you install Intellect and are also available from the Help menu.
- The best source for Intellect training is the DVTTraining.com. DVTTraining.com is a training website that can be used for self-paced learning. It is a free tool that users can use 24 hours a day, 7 days a week at no charge.

NOTE: This guide assumes users are familiar with Microsoft Windows®.

Chapter 1 - Getting Started

This chapter contains important information for both new and experienced users. New users will benefit from general ideas about machine vision as well as from pointers to other sources of information and training. This chapter starts with an introduction to the DVT Vision Sensor product family, including technical specifications and a brief description of every Vision Sensor in that family including the DVT 5X5 Vision Sensors, Line Scan Sensor, Extreme Scanner, and the DVT SC (DVT Spectral Camera). It then moves on to how to install the software (Intellect User Interface), walks the user through the process of setting up the Vision Sensor, and explains how to establish communications with it. The chapter ends with a brief description of the new features of Intellect 1.3, and a guide to other resources.

What is Machine Vision?

Machine vision, in layman's terms, is exactly that. It can be defined as machines with their own set of eyes. Using their vision, machines inspect products for defects by detecting the presence or absence of parts, taking measurements, reading codes, etc. How it works may seem rather complicated at first glance. However, with a little help from basic optics, it is not. The machine's eye mainly looks for changes in contrast and uses a 2-dimensional plane to do so. Our Vision Tools tell the camera (the machine's eye) how to evaluate changes in contrast and in what direction to evaluate them. The camera then collects this information from the image and evaluates it following the user-defined rules. Because the camera's actions are so dependent upon that change in contrast, it is imperative that you adjust your lighting such that the camera (or machine) can get a consistent and close look at the factory product.

Imaging and Pixels

Every image from the Vision Sensor can be considered an array of pixels. For standard resolution systems, the image consists of 640 columns and 480 rows of pixels (for a total of over 300,000 pixels). For high resolution systems, the image contains 1280 columns and 1024 rows of pixels (for a total of over 1.3 million pixels!). Every pixel in the image can be considered a source of information, so from a standard resolution image, the system obtains over 300,000 data points. For grayscale systems, every pixel provides a numerical value from 0 to 255 which is normalized to values from 0 to 100. This value is the intensity level of that pixel. An intensity value of 0 corresponds to pure black, an intensity value of 100 corresponds to pure white, and the values in between correspond to 99 different shades of gray. For color systems, every pixel provides 3 different intensity values: one for red, one for blue and one for green. These three values make up the RGB content of that pixel. Virtually any color can be represented by using the appropriate coefficient for the base colors (Red, Green and Blue). Using this information, Vision Sensors inspect images to provide the user with feedback regarding the presence or absence of a part, flaw detection, code reading and/or verification, etc.

What is a DVT Vision Sensor?

The DVT Vision Sensor is your hardware tool for assuring quality in manufacturing, gathering data, and providing information to all levels of your enterprise. This information might take the form of device-level data such as inspection values being sent to a PLC or process-level Statistical Process Control (SPC) data. This information might be as generalized as a PASS or FAIL signal or as specific as position feedback to a motion controller. You can use the Intellect software to divide and conquer these tasks. Intellect was designed to be easy to use for anyone.

DVT Vision Sensors Product Family

The DVT Vision Sensor Product Family consists of the 500 family of products as well as the DVT Spectral Sensor. The 500 family of products consists of the following Vision Sensors:

- DVT 515 Sensor (1X): The DVT 515 uses a grayscale CMOS device with a resolution of 640 by 480 pixels. It includes all Intellect tools except the Reader Vision Tools and the Spectrograph (exclusive to the DVT Spectral Sensor).
- DVT 535 Sensor (2X): The DVT 535 Vision Sensor is similar in functionality to the DVT 515 Vision Sensor but delivers more precision by replacing the image acquisition device with a CCD. This camera is better than the DVT 515 system for precision measurement applications and analysis of parts that are rich in detail. This camera has 2X the processing speed of the DVT 515.
- DVT 545 Sensor (4X): This vision sensor consists of a 640 by 480 pixel grayscale CCD and a high speed processor. It combines the quality images provided by a CCD with the power of a fast processor to perform delicate inspections at a very high speed. This vision sensor has 4X the processing speed of the DVT 515.

- DVT 542C Sensor (3X): This is the basic color system. It consists of a color CCD of 640 by 480 pixels which makes the Vision Sensor the best choice for most color applications. This vision sensor has 3X the processing speed of the DVT 515.
- DVT 550 Sensor (6X): fastest DVT vision sensor available. It offers grayscale CCD imaging at 640 by 480 resolution and a high-speed DSP chip. This camera has 6X the processing speed of the DVT 515.
- DVT 554 Sensor (6X): high-speed, high-resolution system. It combines a high-resolution CCD (1280 by 1024 pixels) with a high-speed DSP processor. This Vision Sensor can detect very small pinholes, stains, streaks, scratches, bubbles and other minute details that normally go undetected by other smart cameras and sensors. This camera has 6X the processing speed of the DVT 515.
- DVT 554C Sensor (6X): high-speed, high-resolution, color system. It uses a color CCD with a resolution of 1280 by 1024 pixels and adds color recognition in all major color modes and spaces found in the factory environment including RGB, Optical Density (CMY), CIE LAB ΔE, and CIE LCH ΔEcmc. This camera has 6X the processing speed of the DVT 515.

Every DVT Vision Sensor includes the Intellect software to set up inspections on part presence/absence, positioning, defect detection, precise measurement, 1D and 2D code readers, OCR, Color (can be used for grayscale levels in monochrome systems), and programmable tools to accommodate user needs. Figure 1 shows the DVT 535 Vision Sensor.



Figure 1: Member of The DVT Product Family: the DVT 535 Vision Sensor.

The other two devices available in the DVT Vision Sensor Product Family are:

- **DVT LineScan LS Sensor**: The LS, or LineScan sensor uses a grayscale imager (with resolution of 2048 by 1). As a part moves through its field of view it can quickly acquire a single row at a time without suffering from motion blur due to long exposure times. These rows can then be combined into a single high resolution image or processed one at a time indefinitely. This allows acquisition of very high resolution images and inspection of high speed continuous materials. Unwrapping of curved surfaces, high speed continuous material inspection and high resolution image acquisition of moving parts are the main advantages of using a LineScan sensor.

Inspection of those images is no different than with any of DVT's other Legend series cameras. It has the same form factor as the other DV vision sensors. It should be used with an external light that offers a very high intensity minimizing the exposure time. Additional information can be found later in this manual.

- **The Extreme Scanner XS:** Vision Sensor uses the precision of a CCD to read 1D and 2D codes and characters. It is the only industrial barcode reader to include OCR/OCV capabilities. Reading up to 200 codes per second, the Intelligent Scanner is the leader in speed. It can read the following codes: BC412, PDF417, Codabar, Code 39, Code 128, Interleaved 2-of-5, PharmaCode, UPC/EAN, Code 93, RSS-14, POSTNET, Snowflake 2D code and DataMatrix codes ECC200 and older. It can also perform 1D barcode grading, Optical Character Recognition, and Optical Character Verification (OCV). This vision sensor shares the form factor of the Legend 500 Series of vision sensors.
- **The DVT Spectral Sensor**: This Vision Sensor uses a grayscale CCD (with resolution of 640 by 480) and an external spectrograph. It is a spectral line imaging Vision Sensor. The DVT Spectral Sensor captures a line image of a target; using prisms and grating optics, it disperses light from the line image along the spectrum. Each captured image contains line pixels in spatial axis (Y axis) and spectral pixels in spectral axis (X axis). Unlike other color smart cameras using RGB platforms, the DVT Spectral Sensor offers more detailed color information across the entire visible and Near Infrared (NIR) spectrum. This allows the CCD to capture more data for more accurate measurements and detection of very slight color variations. Unlike point color spectrophotometers, the DVT Spectral Camera provides spatial resolution and simultaneous measurement across a large number of points. Figure 2 shows the DVT Spectral Sensor. It shares has the same form factor as the other DVT Vision Sensors but it requires a different piece of hardware to visualize the spectrum of the part being inspected. It should be noted that the DVT Spectral Sensor does not come with the integrated ring light. It uses an external white light that offers a very high intensity minimizing the exposure time.



Figure 2: DVT Spectral Sensor.

DVT Vision Sensor Specifications

Size: 112mm x 60mm x 30mm (not including spectrograph for the DVT Spectral Sensor which adds an additional 122mm to the 30mm dimension or a lens) + an additional 50mm cable clearance added to 112mm

Mounting: Four M4 threaded holes (on back of unit), 7.9mm depth

Weight: 170g / 6 oz. (without lens)

Power Requirements: A regulated and isolated 24 V DC power supply is needed (sold separately): 210mA for the 520 and 530 systems (a minimum 5W) or 300mA for the rest of the Legend Series (a minimum of 7.5 W.) If DVT LED array lights are used, the power supply must provide an additional 30W required per array.

Operating Temperatures: 0-45° C (32-113° F), non-condensing environment; Maximum 50° C / 122° F

Electronic Shuttering: 10μs – 1s exposure times

Optics: CS mount standard, C mount-capable with an adapter (sold separately)

External Ports: Keyed 10 pin RJ-45 connector (Power and Digital I/O), RJ-45 (10/100 megabit Ethernet communications, TCP/IP protocol connector)

Digital I/O: 24 Volts DC regulated, 8 configurable inputs & outputs, NPN (current sinking) inputs, PNP (current sourcing) outputs, active high signals. Inputs can sink up to 1.5mA and outputs can source a maximum of 50mA.

Warning: Users are strongly encouraged to the DVT Digital I/O and Power Cable (sold separately).

Certifications: CE certified

Summary of Image Sensor Information

Table 1. DVT Vision Sensor Information

System	Speed	Description	Resolution	CCD Size (min lens format)	RAM[1]	Flash[2]
DVT 515	1X	Low-Cost Grayscale CMOS (All tools except Readers)	640x480	CMOS 3.8x2.88mm mm (1/4")	64 MB	16 MB
DVT 535	1.5X	Grayscale CCD	640x480	4.8x3.6 mm (1/3")	64 MB	16 MB
DVT 545	4X	High Speed Grayscale CCD	640x480	4.8x3.6 mm (1/3")	64 MB	16 MB
DVT 542C	3X	High Speed Color CCD	640x480	3.2x2.4mm (1/4")	64 MB	16 MB
DVT 550	6X	Highest Speed Grayscale CCD	640x480	4.8x3.6 mm (1/3")	128 MB	16 MB
DVT 554	6X	High Speed High Resolution Grayscale CCD	1280x1024	6.4x4.8 mm (1/2")	128 MB	16 MB
DVT 554C	6X	High Speed High Resolution Color CCD	1280x1024	6.4x4.8 mm (1/2")	128 MB	16 MB
DVT XS (Extreme Scanner)	6X	550 based High Speed Camera Reader (1D Codes, 2D Codes, OCR)	640x480	4.8x3.6 mm (1/3”)	128 MB	16 MB
DVT LS (Linescan)	6X	High Speed, High Resolution Grayscale CCD	2048x1	14.3mmx7.0um (1”)	128 MB	16 MB
DVT SC (DVT Spectral Sensor)	3X	Spectrum analyzer for visible and near IR wavelengths	640x480	4.8x3.6 mm (1/3")	64 MB	16 MB

[1] In RAM memory is where all processing is done. At all points in time almost all data, including all products, reside in RAM.

[2] In flash memory is where data that needs to be persistent (will not be lost upon power down and power up) is stored. It is difficult to estimate the number of Products or Vision Tools that can be stored in a particular Vision Sensor even knowing the amount of memory available. This is because different Vision Tools occupy a different amount of memory. Most Vision Tools that do not require training or learning of a model, template, font list or other parameters take up less than 5 KB. However the exact amount of memory may also depend on the size of the Vision Tool especially for the area-based Vision Tools. The user can look at the amount of flash memory left in a Vision Sensor in the “Product Management” dialog box under the “Products” menu. In Intellect, flash memory is reclaimed automatically.

DVT Vision Sensor Dimensions

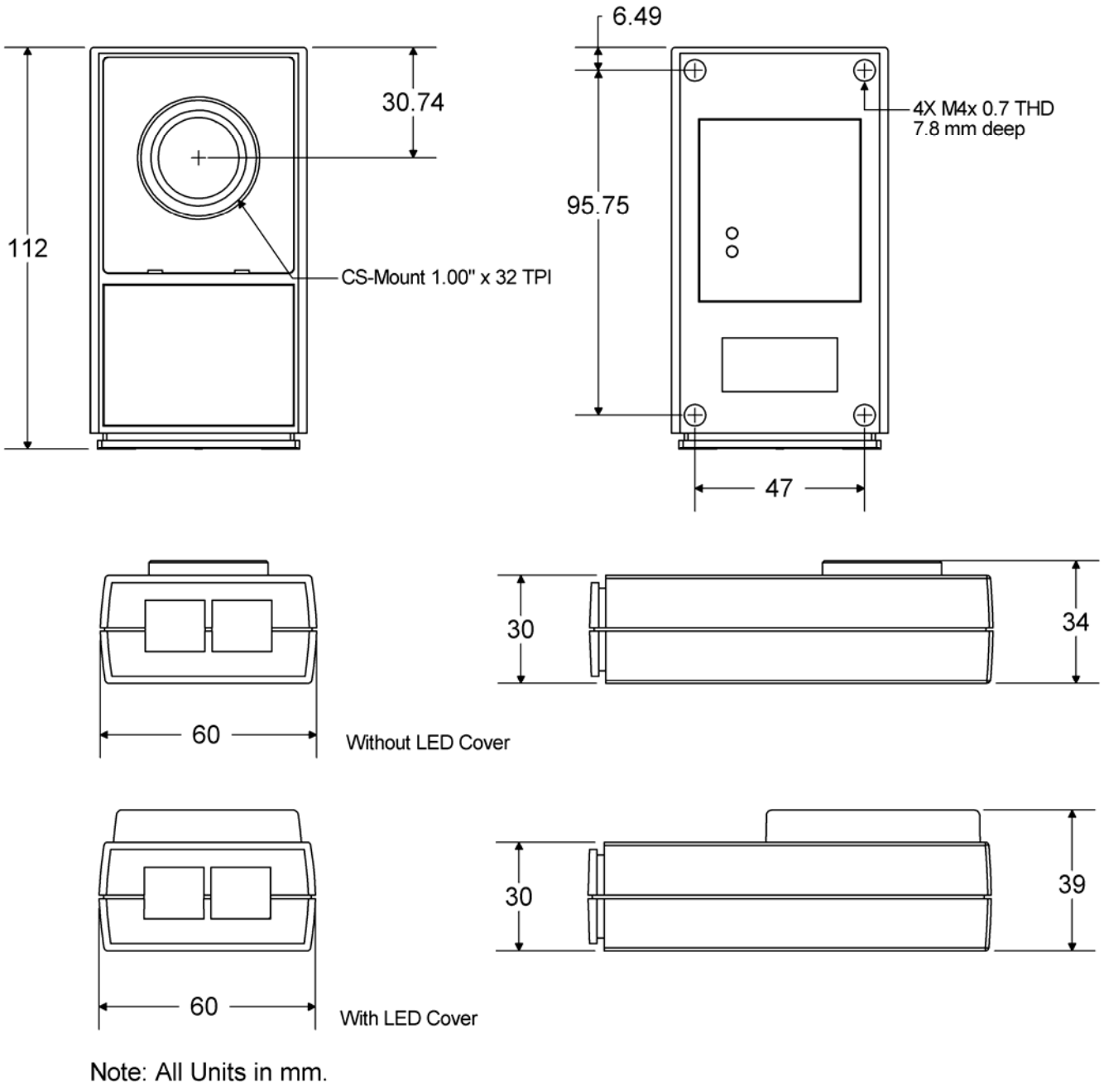


Figure 3. Schematic of DVT Vision Sensor

Installing Intellect Software

While it may be very tempting to immediately insert the Intellect CD into your computer, please take a moment to ensure that installation will go smoothly. The following is a list of requirements necessary for a proper installation of this version of Intellect.

PC Requirements

- 400 MHz Pentium-class PC: For best results, we recommend that you use Windows XP® or a more recent version of Windows® on your PC. However Windows 2000® will work as well. **NOTE**: Windows 98® and Windows NT® 4.0 will not work with Intellect. A minimum of 128 MB RAM will be required. A faster processor will directly relate to faster updates of images in the Intellect user interface.
- Available Space: Intellect requires a minimum 100 MB hard disk space.
- Ethernet Card: 10/100 megabit Ethernet card with integrated RJ-45 jack. Integrated Ethernet is recommended.

Installing the Intellect User Interface

The steps to follow are:

- Insert the Intellect CD. The Intellect installation should automatically start. However, if the CD does not automatically run, follow these steps:
- From the Start menu, choose Run.
- Type D:\setup.exe (substitute the appropriate drive letter of your CD-ROM drive for D:) and press Enter.
- Follow the installation instructions on screen.

After installation, Intellect will reside in the "C:\Program Files\Cognex\DVT\intellect1x\" directory (assuming the user accepts the default selection for directory location). The Intellect User Interface file is named "intellect.exe" and firmware files end with a numerical extension indicating the DVT series number of *.5x5, *.54x, or *.55x (the hardware platform with which they are associated). For example, the firmware file for Intellect 1.3 release 1 for a DVT 515 sensor is named "intellect131.5x5." These firmware files contain the code that needs to be downloaded to the Vision Sensor. They are the part of Intellect that resides in the Vision Sensor where all the calculations take place.

Please take a moment to fill out the registration card included with this manual and return to Cognex. Return of this registration card activates your Warranty.

Connecting to the DVT Vision Sensor

The DVT Vision Sensors can use Ethernet and the TCP/IP and UDP/IP communication protocols. This section outlines how to establish communications with the Vision Sensor. In order to communicate with a Vision Sensor, you will need the following:

- PC: A Pentium class PC with the following hardware and software:
    - Windows® Operating System: Windows XP or 2000. (Windows 2000 or a more recent release is recommended.)
    - Intellect Software: DVT's Intellect User Interface.
    - Ethernet Network Card: 10/100 megabit Ethernet card. The Ethernet card's appropriate drivers should be installed and the TCP/IP communications network protocol should be installed and setup.

- Ethernet Switch (Hub): Full Duplex 10/100 megabit Ethernet Switch is necessary for connecting multiple DVT Vision Sensors in a network.
- Ethernet Cable: Category 5 Ethernet Cable with RJ-45 connectors. (Note: a cross-over Ethernet cable should be used when connecting directly to a single Vision Sensor that does not have a 5x5 part number.)

The RJ-45 connection on your DVT Vision Sensor is designed to communicate between the Vision Sensor and a PC running Intellect. To do this, the PC must have an Ethernet network card running the TCP/IP protocol. Figure 4: shows where the connectors are located in the Vision Sensor.

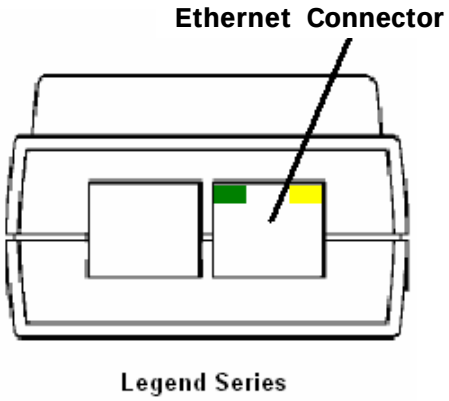


Figure 4: For the DVT Vision Sensor, the RJ-45 (Ethernet) port is the port on the right with two indicator LEDs.

Assigning an IP Address to a Vision Sensor

The PC and Vision Sensor communicate using the TCP/IP and UDP/IP networking protocols. TCP/IP stands for Transmission Control Protocol / Internet Protocol and is based on identifying elements on a network by unique numbers, called IP addresses. In order for two devices to communicate they must reside on the same Local Area Network (LAN), be part of the same sub network, and have unique IP addresses. UDP stands for User Datagram Protocol and is lightweight, connectionless and efficient but it does not guarantee successful and error-free delivery of data packets. This protocol is used within Intellect for browsing the network and finding DVT Vision Sensors.

All DVT Vision Sensors come with a default IP address, which must be changed to a number compatible with both your computer and LAN. An IP address is the equivalent to a telephone number. The Vision Sensor will assign itself a default IP address. You must then re-assign a new IP address that is compatible with your local network. In other words, you must make sure that you have the right phone number to call the Vision Sensor! When the Vision Sensor is in Diagnostics mode, the user can double-click (or right click and choose Setup IP Address) on it to open the Setup Camera IP Address Wizard which will walk the user through this process and make recommendations as to which IP address to use. When the camera is not in diagnostics mode, the user can select a Vision Sensor and enter an IP Address for the Vision Sensor in the Properties Window that appears when the Vision Sensor is selected.

Please use the following steps to set up your computer and DVT Vision Sensor with the correct IP addresses:

- Start Intellect.
- The Network Window will open automatically at startup. See Figure 5.

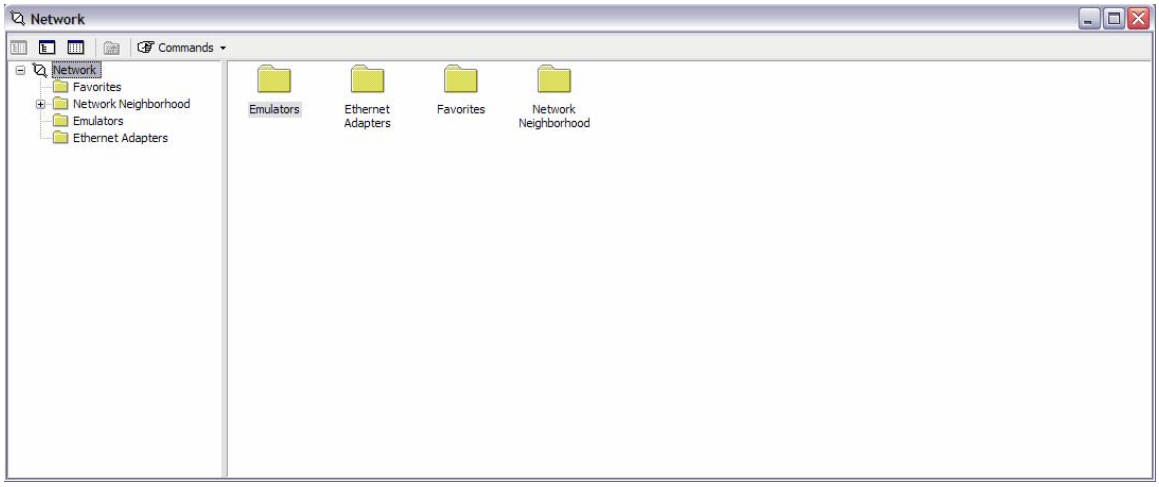
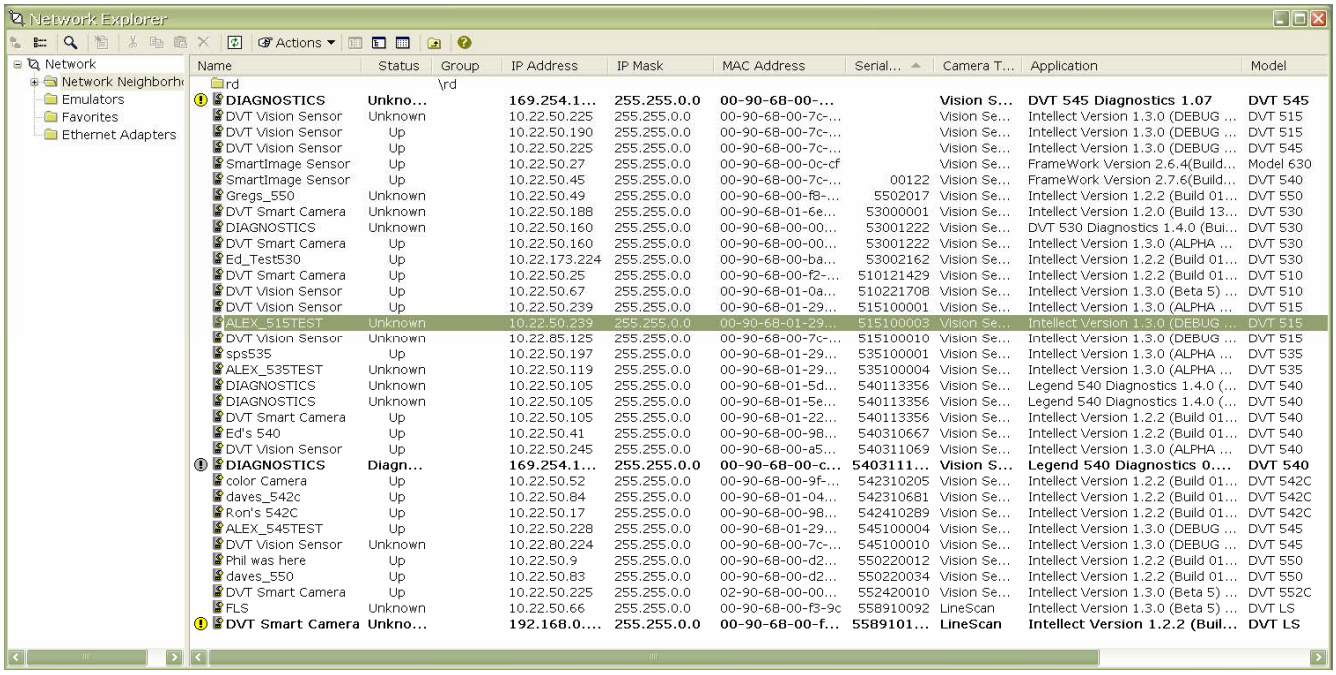


Figure 5: Network Explorer Window

- Click and Open the "Ethernet Adapters" folder. This will show you the current Ethernet Devices that your PC has available, and their properties. Record your computer's (Host) IP Address and IP Mask on a piece of paper. You will need to set the Vision Sensors address to be compatible to your PC's address.
- Click on Network Neighborhood. See Figure 6.
- Network Neighborhood allows you to identify DVT Vision Sensors on a network. Your computer will search your local network for Vision Sensors. If no sensors are found, wait about 10 seconds and search again. There is sometimes a small delay due to the time it takes for your computer to recognize the Vision Sensor.



Name	Status	Group	IP Address	IP Mask	MAC Address	Serial...	Camera T...	Application	Model
rd		\rd							
DIAGNOSTICS	**Unkno...**		**169.254.1...**	**255.255.0.0**	**00-90-68-00-...**		**Vision S...**	**DVT 545 Diagnostics 1.07**	**DVT 545**
DVT Vision Sensor	Unknown		10.22.50.225	255.255.0.0	00-90-68-00-7c-...		Vision Se...	Intellect Version 1.3.0 (DEBUG ...	DVT 515
DVT Vision Sensor	Up		10.22.50.190	255.255.0.0	00-90-68-00-7c-...		Vision Se...	Intellect Version 1.3.0 (DEBUG ...	DVT 515
DVT Vision Sensor	Up		10.22.50.225	255.255.0.0	00-90-68-00-7c-...		Vision Se...	Intellect Version 1.3.0 (DEBUG ...	DVT 545
SmartImage Sensor	Up		10.22.50.27	255.255.0.0	00-90-68-00-0c-cf		Vision Se...	FrameWork Version 2.6.4(Build...	Model 630
SmartImage Sensor	Up		10.22.50.45	255.255.0.0	00-90-68-00-7c-...	00122	Vision Se...	FrameWork Version 2.7.6(Build...	DVT 540
Gregs_550	Unknown		10.22.50.49	255.255.0.0	00-90-68-00-f8-...	5502017	Vision Se...	Intellect Version 1.2.2 (Build 01...	DVT 550
DVT Smart Camera	Unknown		10.22.50.188	255.255.0.0	00-90-68-01-6e...	53000001	Vision Se...	Intellect Version 1.2.0 (Build 13...	DVT 530
DIAGNOSTICS	Unknown		10.22.50.160	255.255.0.0	00-90-68-00-00...	53001222	Vision Se...	DVT 530 Diagnostics 1.4.0 (Bui...	DVT 530
DVT Smart Camera	Up		10.22.50.160	255.255.0.0	00-90-68-00-00...	53001222	Vision Se...	Intellect Version 1.3.0 (ALPHA ...	DVT 530
Ed_Test530	Up		10.22.173.224	255.255.0.0	00-90-68-00-ba...	53002162	Vision Se...	Intellect Version 1.2.2 (Build 01...	DVT 530
DVT Smart Camera	Up		10.22.50.25	255.255.0.0	00-90-68-00-f2-...	510121429	Vision Se...	Intellect Version 1.2.2 (Build 01...	DVT 510
DVT Vision Sensor	Up		10.22.50.67	255.255.0.0	00-90-68-01-0a...	510221708	Vision Se...	Intellect Version 1.3.0 (Beta 5) ...	DVT 510
DVT Vision Sensor	Up		10.22.50.239	255.255.0.0	00-90-68-01-29...	515100001	Vision Se...	Intellect Version 1.3.0 (ALPHA ...	DVT 515
ALEX_515TEST	Unknown		10.22.50.239	255.255.0.0	00-90-68-01-29...	515100003	Vision Se...	Intellect Version 1.3.0 (DEBUG ...	DVT 515
DVT Vision Sensor	Unknown		10.22.85.125	255.255.0.0	00-90-68-00-7c-...	515100010	Vision Se...	Intellect Version 1.3.0 (DEBUG ...	DVT 515
sps535	Up		10.22.50.197	255.255.0.0	00-90-68-01-29...	535100001	Vision Se...	Intellect Version 1.3.0 (ALPHA ...	DVT 535
ALEX_535TEST	Unknown		10.22.50.119	255.255.0.0	00-90-68-01-29...	535100004	Vision Se...	Intellect Version 1.3.0 (ALPHA ...	DVT 535
DIAGNOSTICS	Unknown		10.22.50.105	255.255.0.0	00-90-68-01-5d...	540113356	Vision Se...	Legend 540 Diagnostics 1.4.0 (...	DVT 540
DIAGNOSTICS	Unknown		10.22.50.105	255.255.0.0	00-90-68-01-5e...	540113356	Vision Se...	Legend 540 Diagnostics 1.4.0 (...	DVT 540
DVT Smart Camera	Up		10.22.50.105	255.255.0.0	00-90-68-01-22...	540113356	Vision Se...	Intellect Version 1.2.2 (Build 01...	DVT 540
Ed's 540	Up		10.22.50.41	255.255.0.0	00-90-68-00-98...	540310667	Vision Se...	Intellect Version 1.2.2 (Build 01...	DVT 540
DVT Vision Sensor	Up		10.22.50.245	255.255.0.0	00-90-68-00-a5...	540311069	Vision Se...	Intellect Version 1.3.0 (ALPHA ...	DVT 540
DIAGNOSTICS	**Diagn...**		**169.254.1...**	**255.255.0.0**	**00-90-68-00-c...**	**5403111...**	**Vision S...**	**Legend 540 Diagnostics 0....**	**DVT 540**
color Camera	Up		10.22.50.52	255.255.0.0	00-90-68-00-9f-...	542310205	Vision Se...	Intellect Version 1.2.2 (Build 01...	DVT 542C
daves_542c	Up		10.22.50.84	255.255.0.0	00-90-68-01-04...	542310681	Vision Se...	Intellect Version 1.2.2 (Build 01...	DVT 542C
Ron's 542C	Up		10.22.50.17	255.255.0.0	00-90-68-00-98...	542410289	Vision Se...	Intellect Version 1.2.2 (Build 01...	DVT 542C
ALEX_545TEST	Up		10.22.50.228	255.255.0.0	00-90-68-01-29...	545100004	Vision Se...	Intellect Version 1.3.0 (DEBUG ...	DVT 545
DVT Vision Sensor	Unknown		10.22.80.224	255.255.0.0	00-90-68-00-7c-...	545100010	Vision Se...	Intellect Version 1.3.0 (DEBUG ...	DVT 545
Phil was here	Up		10.22.50.9	255.255.0.0	00-90-68-00-d2...	550220012	Vision Se...	Intellect Version 1.2.2 (Build 01...	DVT 550
daves_550	Up		10.22.50.83	255.255.0.0	00-90-68-00-d2...	550220034	Vision Se...	Intellect Version 1.2.2 (Build 01...	DVT 550
DVT Smart Camera	Up		10.22.50.225	255.255.0.0	02-90-68-00-00...	552420010	Vision Se...	Intellect Version 1.3.0 (Beta 5) ...	DVT 552C
FLS	Unknown		10.22.50.66	255.255.0.0	00-90-68-00-f3-9c	558910092	LineScan	Intellect Version 1.3.0 (Beta 5) ...	DVT LS
DVT Smart Camera	**Unkno...**		**192.168.0....**	**255.255.0.0**	**00-90-68-00-f...**	**5589101...**	**LineScan**	**Intellect Version 1.2.2 (Buil...**	**DVT LS**

Figure 6: Network Neighborhood

- Highlight the DVT Vision Sensor with the appropriate serial number. In the properties window on the right side of the screen is where you can edit the Name, IP Address and IP Mask. Set the IP address to be same as your computer's IP address except for the last number. For instance, Computer: 192.168.1.113; DVT Vision Sensor: 192.168.1.32. If you are not sure about which IP address to use you should ask your network administrator. See Figure 7.

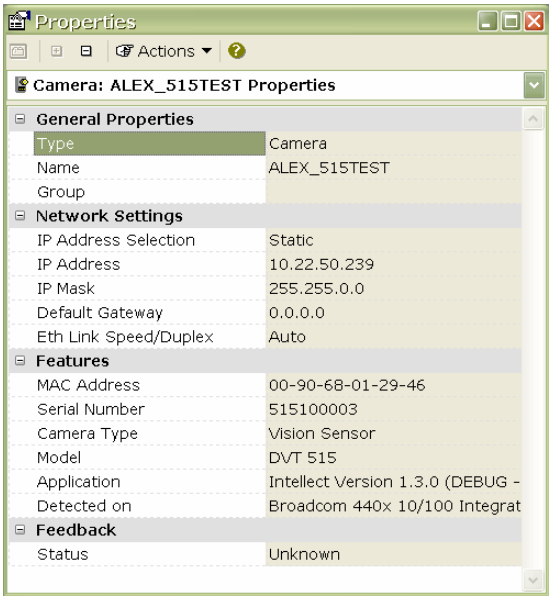


Figure 7: Properties Window

- If the IP address is already being used by another device, The software will return to the original address . Simply go back and edit the systems' IP address once more to a number that is not currently being used. An icon will appear next to the name of the camera if it is unreachable or in Diagnostics.

Hint: After configuring your DVT Vision Sensor with its new IP address, you should affix a label to the Vision Sensor with the Ethernet properties so that you do not forget this important information.

Repeat these steps for each DVT Vision Sensor you have. When complete, the PC and all sensors will have unique IP addresses. Connect all of the systems to a hub (or multiple hubs) and the hub to your PC. You now have a "DVT network."

- For more advanced network configurations, you will need to adjust the Sensor Address Mask. For a DVT Vision Sensor and PC to communicate, they must be on the same sub-network. This means that the masked portions of the sensor's IP address must be identical to the PC's. For example, with the default mask of 255.255.255.0 on your Vision Sensor, only the fourth portion of the PC's IP address may be different from your Vision Sensor. In this example, a sensor at 192.168.0.242 can communicate with a PC at 192.168.0.200, but not with a PC at 192.168.1.200.

The following table summarizes the use of the Sensor Address Mask property with Ethernet networks.

Table 2. Network Address Configurations

Sensor Address Mask	Sensor IP Address	Valid Sample PC IP Addresses
255.255.255.0	192.168.0.242	192.168.0.200, 192.168.0.100, 192.168.0.240
255.255.0.0	192.168.0.242	192.168.1.200, 192.168.35.67, 192.168.222.35
255.0.0.0	192.168.0.242	192.25.1.200, 192.222.22.242, 192.0.1.45

Warning: Whenever connecting a node to an existing network, please exercise extreme caution. Connecting a DVT Vision Sensor to a network without the Network Administrator's knowledge is not recommended.

DVT Vision Sensor Hardware Setup

In this section, physical connections to the DVT Vision Sensor are explained. This includes powering the system, connecting the communications cable, and connecting to the I/O breakout board.

Unpacking and Powering the DVT Vision Sensor



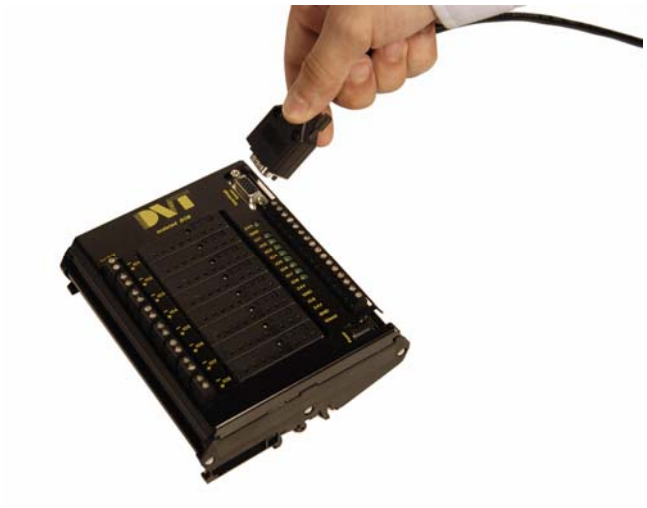
Connect the keyed RJ-45 Digital I/O and power cable to the DVT Vision Sensor (typical part number: CBL5-D10).

Note: The cable used for The DVT Vision Sensor has a male 15-pin high density DB-sub-connector on one end and a keyed 10 pin RJ-45 connector on the other end.

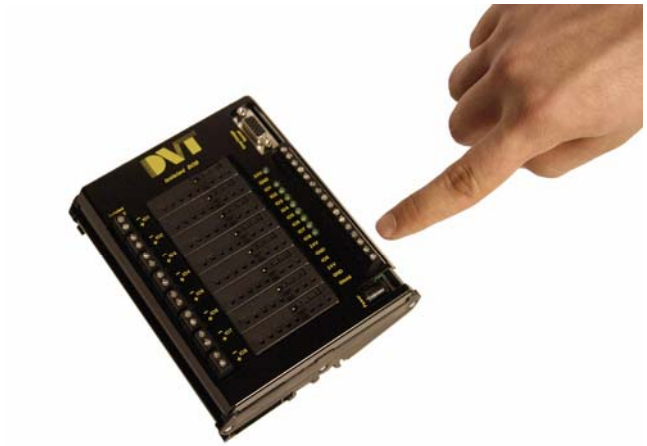


Connect the RJ-45 Ethernet cable (typical part number: CBL-C10E).

Note: If you are connecting the other end of the Ethernet cable directly to a PC (as opposed to a hub), make sure to use either a crossover Ethernet cable (CBL-C10EX) or a crossover connector (CON-RJ45N).



Connect the opposite end of the digital I/O & power connector (High Density 15 pin D connector [CBL5-D10]) to the DVT I/O breakout board (CON-IBOB). Or, if you do not plan on using the breakout board, cut the connector from the 15-pin cable and wire directly into power and I/O destinations.



Connect a 24V DC power supply to the DVT I/O breakout board using the last three screw-down terminals.

Note: For laboratory use, a power supply is available from DVT. The part number is ACC-24-NA (for North America). This power supply uses the power jack on the DVT I/O breakout board.



The isolation breakout board contains two sets of terminals: one that is directly connected to the Vision Sensor, and one that allows for the use of opto-isolation modules. As we saw in the previous steps the port on the right (see diagram) includes I/O, strobe and power screw-down terminals. The I/O lines on this port are to be used when isolation is not needed.

The second port, the one on the left side (see diagram) has the I/O connections to be used in conjunction with isolation modules. These terminals are inactive unless isolation modules are used. Users should decide which lines need isolation and install the corresponding modules (input or output) for those lines only. For more information refer to the following document on the DVT website: DVT Isolation Break Out Board Note.



Apply power to the Vision Sensor and wait for the system to boot. The Status light will illuminate with a blinking green light when the boot sequence is finished. The blinking green light will always blink when the camera is running and will increase the blink rate as the load increases.



Attach a lens to your Vision Sensor. If you are using a C-mount lens, you will find it necessary to use a 5mm adapter (LDS-C). If you have a CS-mount lens, you will not need the adapter.

Getting More From Intellect

In addition to the Installation and User's Guide, you can learn more about Intellect software from the following resources:

On-line Documentation & Help Files

After Intellect Software has been installed on a PC, all the necessary documentation is included along with the application software. HTML Help and PDF (Portable Document Format) versions of the written documentation are included.

- HTML Help: The help file is a great source of information about all aspects of Intellect. There are several ways to access the help file:
    - Click on the Help menu and choose Help File Index or press F1.
    - Use the Help buttons located in most dialog boxes (navigates to the relevant location in the help file).
    - Hold the Shift key and press the F1 key to get the  prompt. From this, click on any part of the user interface for context-sensitive help.
    - The Help file can be viewed independent of a Intellect session. Simply click on the Windows® Start menu, choose DVT Applications, and then choose "Intellect Help".
    - Access them directly from the DVT website. These help files are continuously updated by DVT.
- PDF Files: The written documentation provided by DVT in PDF format includes this Installation & User Guide. In addition, the DVT Script Reference Manual is both loaded on your computer and available on-line in PDF format. A hard copy of the Script Reference can also be purchased separately through a DVT Distributor.
- Intellect Seminars/Workshops: Cognex offers Camera products through a worldwide network of Automation Solution Providers (ASPs). Local seminars and workshops can be found on the DVT website. You can find this information on the web under the Education portion of the DVT website. Also, you can contact your local ASP for dates and locations of the next Intellect Seminar/Workshop. These workshops offer an introductory view of both the Camera hardware platform and the Intellect software. The ASP Seminar/Workshop is a perfect way to "get your feet wet" on DVT's inspection systems.
- Intellect Training Courses: Comprehensive four-day training courses are taught at Cognex's Atlanta and Detroit training facilities as well as other Cognex locations worldwide. These courses are a great way to get a quick start on learning Intellect. They are free of charge, with users responsible only for their transportation, dinners, and lodging. To learn more and to sign up for a class, please contact your local Automation Solution Provider (ASP) or go to http://www.dvtsensors.com/training and click on "Classroom training" for the schedule.
- CD-Rom/Self Paced Training: DVTTraining.com is a website which deliveries training through self paced online training videos and emulator examples. This training is available at www.dvttraining.com and is available for free from Cognex.

- Online Training: Weekly 45-minute courses using the latest technology of online conferencing. These courses are free of charge as well; please contact your distributor for more information.
- Subscribe to the SmartList: Join the long list of Camera users who regularly discuss topics via DVT's SmartList server. Discussion topics include (but are not limited to):
    - Application questions: "What light source should I use?"
    - Intellect software questions: "Which Vision Tool is best for this application?"
    - Integration questions: "Has anyone gotten the Vision Sensor & a XYZ PLC to communicate?"

  For information on SmartList, visit the DVT web site, http://www.dvtsensors.com and click "Support".

  Users can join the SmartList by visiting the support section of the DVTSensors website.
- Cognex Certified Automation Solution Provider (ASP): Cognex sells DVT products through a worldwide network of Automation Solution Providers. These ASPs are certified annually based on in-depth knowledge of both Intellect and the DVT Vision Sensor. If you are unsure of who is your Cognex Solution Provider, you can search for that information on the Cognex website. Please contact your local ASP with your technical support questions.
- Stay informed about New Intellect Releases: Visit the DVTSensors Web site for up-to-date information about forthcoming releases of Intellect. There are two easy ways to reach the web site:
    - From Intellect, click on the Help menu and choose DVT on the Internet.
    - Start your web browser and type http://www.dvtsensors.com on the address bar.

Chapter 2 - A Closer Look at the System

This chapter describes all the basic principles behind DVT Vision Sensors. The internal structure of the Vision Sensor is described and a brief description about the parameters available at every level is provided. From system level (unique to every Vision Sensor) to Vision Tool level (dependent upon the Vision Tool being used), this chapter illustrates the use of their parameters and the functionality they add to the system. Once the user becomes familiar with the internal structure of DVT Vision Sensors, the use of these will become a very simple procedure because Intellect is designed to be an easy to use interface. The DVT Vision Sensor hardware emulator is also explained in this chapter. The hardware emulator is a very powerful tool that simplifies troubleshooting, shortens setup and assists in training.

As mentioned before, Vision Sensors can take an image and analyze it to determine whether, based on user-defined parameters, that is a good image or not. Based on this output from the Vision Sensor, the user can take the necessary action (e.g. reject a part, fix the production process, etc.). The purpose of this section is to discuss the functionality inside the Vision Sensor. Figure 8 shows the hierarchical tree determined by inner components of Vision Sensors.

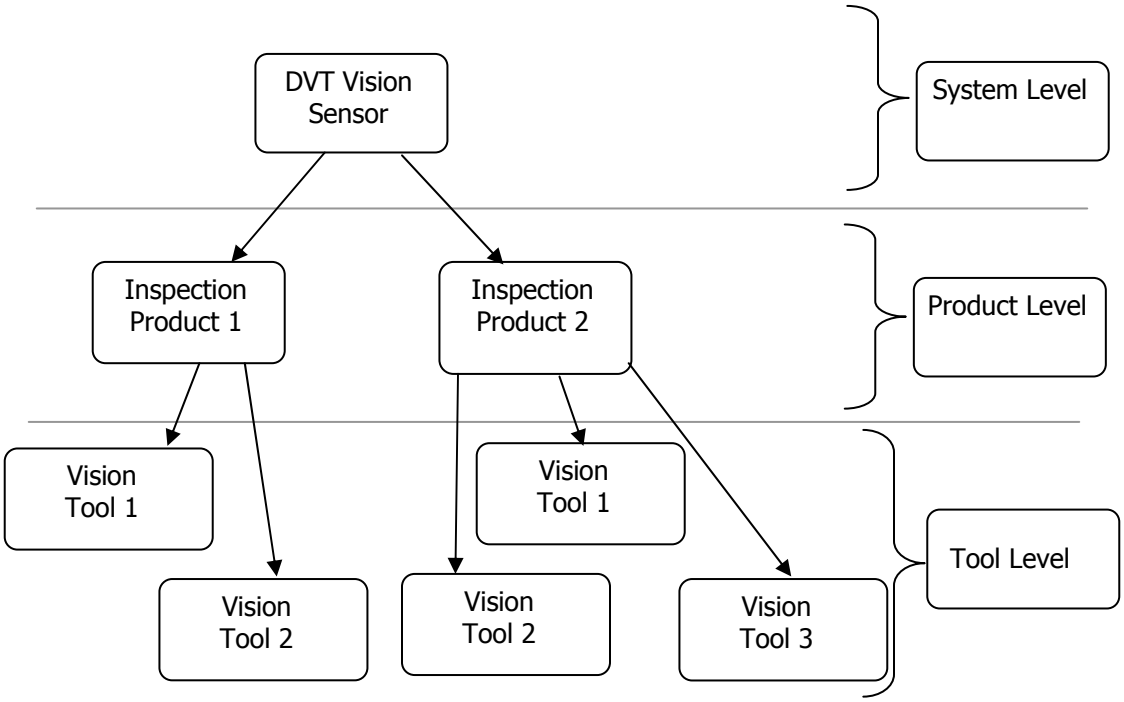


Figure 8. Hierarchical organization within the Vision Sensor.

At the top level, we find the system parameters. These parameters are common to every inspection that the Vision Sensor performs; they affect the overall behavior of the Vision Sensor rather than that of a certain inspection. Next, at the product level, the user can change parameters that affect a specific inspection (such as illumination). Essentially, a product is directly associated with an inspection, so product parameters affect only one of the inspections that the Vision Sensor is performing. Finally, at the Tool level, the user can set some Tool parameters. Each Tool performs part of an inspection; here is where all the pieces of every inspection are defined. Vision Tools are associated with inspection tasks.

In order to illustrate this functionality let us work with an example. The part in Figure 9 needs to be inspected. First, we need to verify that the drillings on the left end are vertically centered, a certain distance apart, and of a certain radius. After that, we need to verify that the label on the right end is centered, aligned and that it contains the correct code. Finally, the specific data from every inspection has to be sent out of the Vision Sensor using Modbus communications. The existence of products inside Vision Sensors simplifies this task. The inspections can be separated into two groups, those that take place on the left end of the part and those that take place on the right end of the part. This will allow the user to trigger the Vision Sensor twice as the part moves in a conveyor from left to right. The first trigger would occur when the left end is in front of the Vision Sensor and the second one when the right end is in front of the Vision Sensor.

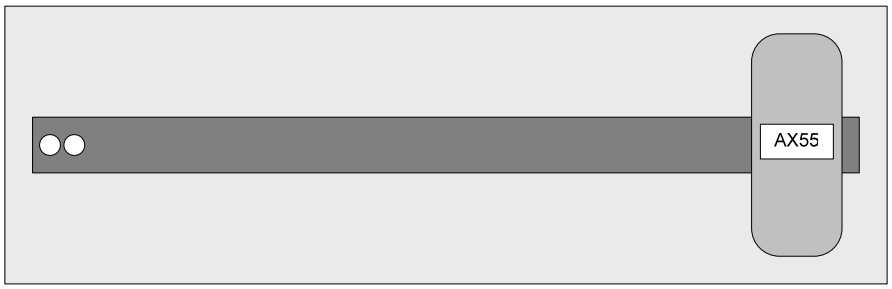


Figure 9: Sample part to be analyzed in two different inspections.

This would not only add modularity to the process but also would allow for a more precise inspection because now we can zoom in on the part. There is not a need to have the entire part in a single image. The inspections are then separated into products. A product would inspect the left end of the part and another product would inspect the right end. In order to determine if there is a problem with the part, those products would use a number of Vision Tools. For the first product, both Math and Measurement Vision Tools are needed to perform the inspection. For the second product, Math and Measurement Vision Tools are again needed and a label reader Vision Tool (OCR Vision Tool) would also be required in this case. Finally, to send out the data, communications, a system level parameter, needs to be configured. Regardless of the inspection taking place, a data transfer will occur.

System Parameters

As mentioned before, the System level of a Vision Sensor controls the overall functionality of the Vision Sensor. The main system parameters are inspection mode, password protection, I/O configuration, background scripts, power-on-product, trigger source, FOV balance, and communications setup.

Inspection Mode

A Vision Sensor can be taking images for the purpose of inspecting them or simply for displaying them in a user interface. It is important to distinguish between these two modes regarding image acquisition: play mode and inspection mode. Play mode applies to each user interface connected to a Vision Sensor and it is independent of the inspection process so its activation does not require the presence of an inspection product. When Real-Time-Feedback is enabled in a user interface, play mode becomes active and the Sensor sends images to be displayed by each program. Inspection mode determines whether the Vision Sensor is currently inspecting images or not. When inspection mode is active, the Vision Sensor sets all configured output signals and sends out any TCP data that the user has set up. The activation of inspection mode requires an inspection product. If a product is selected as the inspection product and the inspection mode is activated, then when the Vision Sensor is triggered, it acquires an image and inspects it according to the Vision Tools that the chosen inspection product contains and sets all appropriate outputs.

Password Protection

In order to provide administrators with access control, Intellect supports the creation of different user accounts. This allows system administrators to give specific control of the system to different users. Each user can be given certain access restrictions, from simply stopping the inspection mode to editing the parameters of every Vision Tool. Administrators can choose which tasks each user is allowed to perform on the system. Furthermore, an inactivity timeout can be set to maximize security. The access control is independent of the PC being used: the access control information is stored in the internal flash memory of the Vision Sensor.

Please be sure to write down your administrator username and password. There is no means of recovering from a secured system without a valid username and password.

Digital I/O Configuration

Vision Sensors have a number of configurable I/O lines. The DVT Vision Sensor contains eight I/O lines. The I/O lines can be configured to better suit the user's needs. Vision Sensors do not have fixed input or output lines. Every I/O line can be configured as an input or as an output depending on the user's needs. This assignment is a system parameter because another device will be connected to the I/O lines, and this cannot change between inspections. It should be mentioned that the number of I/O lines in the DVT Vision Sensor can be increased to 24 I/O points by the DVT remote Ethernet I/O device, CON-IOE.

Input Functions

Inputs on the Vision Sensor can serve one of many functions. The following table summarizes these functions:

Table 3. Input Functions

Input Function	Purpose
Product ID Bits	Each line will represent a digit to make up the binary code that identifies a product based on the assigned Product ID.
Product Select	Toggled when **Product ID** bits are active to select a Product. This signal tells the Vision Sensor that the Product ID in binary code is ready for reading.
Inspection Trigger	Turned on to start acquisition of an image for inspection when the system is in external trigger mode.
Encoder Inputs	Used to time inspection triggers based on encoder inputs. These are labeled 'Event Counter' and 'Quadrature Input'.
Digital Relearn	Used by Positioning, Counting, OCR, Code readers, Spectrograph, and Color Vision Tools to automatically relearn or reset Vision Tool parameters. Relearning will occur if the Digital Relearn input is held active during the next inspection and the Vision Tool has its Digital Relearn parameter checked.
External Trigger Mode	Set the camera to external trigger mode. This means that it will not acquire images unless a trigger signal is sent by an external device.
Result Acknowledge	Informs the camera that the inspections results were received. For Inspection Results under Result Type the user specifies how the camera should output its physical signal (e.g. hold last state until next inspection). One of the options is to maintain the value until this Ack signal is received. This way the user can establish a handshake between the camera and the industrial device communicating with it (e.g. PLC).
Event Counter	This is used when an encoder is connected to the camera to time events based on tick counts rather than absolute time. This is particularly useful in the Line Scan system where this signal will be connected to either of the two output lines of an encoder to receive the line triggers for image acquisition. However, it can also be used for other parameters such as specifying a delay after trigger.

Quadrature Input	This is used in the Line Scan system. It will be connected to either of the two output lines of an encoder to receive the line triggers for image acquisition. This signal is specifically used in combination with Event Counter and will only work if you are using the encoder settings under Physical I/O in Quadrature Counting mode. Only available for Pin 2.
Disable Run Mode	Stops the camera from executing inspections. This does not necessarily mean that the camera will stop acquiring images (if the user interface is in play mode, the camera will still take images. However, it means that the camera will not set any of its outputs.
User Inputs	Inputs 1-32 are used in combination with DVT Scripts and Background Scripts to monitor inputs. (See the Script Reference Manual for more details and examples.)

In addition to assigning specific functions to the I/O pins, the user can specify other parameters to ensure that the inputs are correctly read by the system. The use of these parameters depends on the hardware device providing the connection to the Vision Sensor. These parameters are polarity inversion, debounce times and input filtering. Polarity inversion is used to change the system lines from active high to active low. Debounce times are used when the signal received by the Vision Sensor may not be stable such as when a physical switch is used. If this is the case, the user can select a period of time during which the Vision Sensor does not look at the line after registering a transition in the state of that line. Filtering is used to filter noise spikes that may occur.

Output Functions

There are several output functions that can be assigned to the I/O lines. The following table summarizes the available output functions and their meaning. Any function may be assigned to any I/O line.

Table 4. Output Functions

Output Function	Purpose
Pass	After inspection is completed, **Pass** is set to show that all Vision Tools within the Inspected Product passed inspection.
Fail	After inspection is completed, **Fail** is set to show that any Vision Tool within the Inspected Product failed inspection.
Busy	**Busy** is set after **Inspection Trigger** is read until the inspection result (**Pass** or **Fail**) is signaled.
Strobe 1	When active, the corresponding strobe light source will be on during image exposure.
Strobe 2	When active, the corresponding strobe light source will be on during image exposure.
Strobe 3	When active, the corresponding strobe light source will be on during image exposure.
Resource Conflict	Signals a conflict between inspections trying to use the same system resources. Typically, signals that an **Inspection Trigger** was received while the system was acquiring another image.
Run Mode	Output used to indicate the inspection running state of the system. The output will be active when the system is set to running of inspections.

Trigger Mode	Output used to indicate the inspection running mode of the system. The output will be active when the system is set to external trigger and inactive when system is set to internal trigger.
Trigger Ack	Signal that goes active when the input trigger is detected.
Select Pass	While **Select** input is active, **Select Pass** signals that the **Product ID** bits specified a Product ID found in the system.
Select Fail	While **Select** input is active, **Select Fail** signals that the **Product ID** bits specified an invalid Product ID.
Acquiring	Signaled after **Inspection Trigger** is read, during exposure time and image digitization. Another **Inspection Trigger** cannot be signaled while **Acquiring** is active.
Inspecting	Signaled after the **Acquiring** signal while an image is being processed for inspection. (Note: **Acquiring + Inspecting = Busy**)
Inspection Toggle	Changes state once after each inspection is performed.
User1 – User16	User definable outputs that go active or inactive based on combination of the results of the Vision Tools being used. User Definable outputs are Product-specific.
Wrong Code	Indicates whether any reader Vision Tool in the product failed because of a string mismatch (high) or because it could not read anything (low).
Script Outputs	Script Outputs 1-16 that can be set through Scripts. See the DVT Script Reference Manual for more details.

Unlike for input functions, in addition to assigning specific functions to the I/O pins, the user can specify the polarity of an input. Any line configured as an output on a Vision Sensor system is active in the high state. The "Invert Polarity" option makes it active in the low state.

Other Communications Settings

There are several ways to get data into and out of Vision Sensors including Ethernet and serial communications. For specific information on these methods see Chapter 5 – DVT Vision Sensor Integration.

Background Scripts

Background Scripts are system-wide programs that can run independent of the Inspection Product. Some tasks assigned to Background Scripts are Product selection for an inspection, register reading and writing, exposure time control, communications, etc. To get more information on Background Scripts, consult the DVT Script Reference Manual, available from DVT under the part number DOC-SCR or from the DVT website for free download as part of the Intellect Software.

Power-on Product

The Power-on Product allows the system to begin running inspections as soon as the Vision Sensor is turned on and "alive". Only Products that have been saved to flash memory may be assigned as Power-on Products because those that are only in RAM memory will be lost on power-down. The Power-on Product assignment is critical when the Vision Sensor is used as a stand-alone system. Once the system is configured with a PC, the Power-on Product must be assigned in order for the sensor to start running inspections (thus signal digital outputs) immediately after cycling the power.

When the system has any products saved in flash memory, Intellect will be in Run mode. In selecting the Power-on Product at start-up, the Vision Sensor selects the named Product for inspections (called the Current Inspection Product).

Note: A Vision Sensor without a product saved to flash memory will not start running inspections after power-up.

Trigger Source

Another system parameter is the trigger source. This parameter indicates how the system is triggered. The options are external trigger and internal trigger. External trigger requires that the trigger input function be assigned to an I/O line and that an external device provide a signal on that input whenever an image is to be taken. Internal trigger is an automatic mode, in which the Vision Sensor takes a new image periodically at fixed user-defined intervals or at an automatic rate. LineScan systems have an extra mode for continuous mode acquisition used for web applications.

Note: When working at full speed the Vision Sensor may show a slow response to communications because the inspections take priority over other processes.

When external trigger is the option for triggering the system, the user can define a fixed delay after trigger if the detection of the part to be inspected cannot be made at the precise moment that the part is in front of the Vision Sensor. This way, the user can have a Tool detecting the position of the part, and triggering the system, but the Vision Sensor will not acquire an image until the part is precisely in the correct position.

FOV Balance

FOV balance includes two different operations: White Image Normalization and Dark Image Subtraction.

White Image Normalization is the process that was known as FOV balance in previous releases. When the illumination is not uniform and the physical setup cannot be improved, doing a White Image Normalization may be helpful. The user can select to calibrate (automatic process) and activate a white normalization of the field of view in a certain region of the image (or over the entire image). This will correct the pixel intensities to make the illumination seem even and well-distributed across the image or the selected region. In general, uneven light should be made even by changing the physical setup (position, source, etc.) instead of using this option. This procedure is not recommended for precision measurement applications, in which the information contained in every pixel may be vital for the success of the inspection.

The Dark Image Subtraction, available on all cameras, is generally used in applications where the exposure time is high. When this is the case, some regions of artificially high intensity levels may appear due to certain characteristics of the CCD. To compensate for this occurrence, Dark Image Subtraction may be applied by covering the lens with a lens cap and pressing "Calibrate". This calibration can then be used in the Spectrograph Vision Tool by selecting the option "Subtract Black Image". This will eliminate the aforementioned unwanted effect by subtracting the learned black image from the live region of interest of the Vision Tool. This process does not affect the original image since it is applied just for the processing of a particular Vision Tool

Dark Image Subtraction functionality can also be used to subtract two images from each other. In these cases, the learned image does not need to be dark. In the "Subtract Dark Image" filter uncheck "Truncate Negative Values" to get the absolute value difference between two images.

Sensor Gain

The sensor gain adjustment makes your Vision Sensor's CCD more sensitive to light. Gain has a maximum value of 25 and is initially set to 2. A Gain of approximately 2 generally works best for the Vision Sensor.

Hint: Is your application on a constantly moving conveyor line? Effective use of the gain setting allows you to minimize blur in your images due to relatively long exposure times. With parts moving past the sensor, increase the gain, then decrease the exposure time until the images no longer appears blurry.

Reflectance Calibration

The goal of Reflectance Calibration is to have the measured reflectance of the sensor match an external device. Color monitoring and 1-D ANSI barcode grading are the Vision Tools that can benefit from this. Note that Reflectance Calibration can be performed for gray scale sensors as well as color.

There are two reasons why you may need to have Reflectance Calibration for color sensors. One reason is to have more accurate relative color measurements with Color Monitoring. The absolute color values themselves would not be accurate but the relative ones are closer to reality. Another reason why you may need Reflectance Calibration is to match two sensors to each other. If they are both calibrated with a single target, the color values they measure will match. For more information on this procedure refer to the "Integration Notes" section of the DVTSensors website or the Intellect help file.

Spectrum Calibration

Spectrum Calibration is only available for the DVT Spectral Sensor. It is the process through which pixel data is translated into wavelength information. When working with the DVT Spectral Sensor sometimes pixel information is not descriptive enough, so this calibration option is available. The process will involve the use of a part or illumination source with known peak wavelengths. After the pixel location and wavelength is entered for each of these peaks, the system interpolates them to obtain a transformation model for each pixel in the region of interest. After the calibration is done, it will be used by all Spectrograph Vision Tools. For more information on this procedure refer to the Intellect help file. "Auto Calibrate With He/10 Lamp" is used in manufacturing with a known lamp.

Product Parameters

The next hierarchical level inside Vision Sensors is the Product Level. At the product level, we have independent subsystems that are responsible for specific inspections. These subsystems are called products. All the products in a system use a common set of system parameters (as discussed before). Since every product is responsible for a specific inspection, products need to have control over the imaging process. That is why parameters such as illumination, image window, and exposure time are Product Level parameters.

Exposure Time

Exposure time refers to the time that the electronic shuttering mechanism exposes the image-acquiring device to light. The longer the exposure time, the more light enters the device. Three factors will determine the exposure time you set for an inspection:

- The overall speed of the parts. For high-speed parts, short exposure times are recommended to minimize blur in the images.

- The overall inspection rate (parts per minute). In situations where the overall part rate is high, exposure time should be minimized to optimize the inspection.

- The available light. Brighter lighting fixtures enable shorter exposure time whereas dimmer ones require longer exposure time.

There may be instances in which you must reduce your exposure time. You have the following options to help reduce exposure time and still maintain overall image intensity:

- Increase the light aimed at the inspection area.

- Increase the gain on the CCD. Gain is the ratio of output over input, or, in this case, the overall increase in image brightness. Increasing the gain makes the CCD more sensitive to light. **Caution: high gain settings may produce a grainy image, which may affect sensitive inspections**.

- Use a lens with an adjustable aperture. Many C and CS mount lenses have aperture controls.

Notes: An aperture is the hole that lets light through the lens onto the imaging surface. The larger the hole, the more light can strike the CCD. Opening the aperture may reduce the depth of field. Doing so will make it more difficult to focus on objects if they vary in distance from the lens.

Increasing the exposure time slows down image acquisition process and will increase image blur for moving inspections.

Digitizing Time

This parameter determines the precision of the analog to digital conversions in the image acquisition device. The slower digitizing time allows for more precise conversions. It is recommended to use the default value. Visible changes in the image are not likely to be observed when changing this option. Depending on the system, users can choose between different digitizing times to speed up the process or to get the best possible image. A parameter directly related to the digitizing time is the partial window acquisition (explained below). When partial image acquisition is being used, the acquisition time decreases to a fraction of the selected digitizing time. This fraction is proportional to the size of the image being acquired.

Partial Image Window

Some applications may not need to process the entire image, but just a portion of it (if the part to inspect is consistently located). Vision Sensors allow the user to select a specific window within the image so only that window is acquired.

One portion of the total inspection time is the image acquisition. The Vision Sensor CCD, with a resolution of 640x480 pixels takes approximately 13ms to transfer the image from the CCD into memory. This time is proportional to the number of rows being acquired. Reducing the acquisition window to 320x240, for example, would cut the acquisition time to approximately 1/2 the original value.

Note: For some imaging sensors, the area to the left of the desired partial image must also be transferred as seen in the figure below. Therefore, to achieve the fastest image transfer, choose a partial image region that has the least amount of rows and is as far left as possible.



Figure 10: Partial acquisition window for 535 and 545 systems.

Antiblooming

Blooming effect occurs when pixels absorb so much light that they saturate and cause neighbor pixels (mostly in the vertical direction) to saturate even though their original intensity level was below saturation. When Antiblooming is turned on, local spots of overexposure are reduced, thus minimizing streaks and washed out areas of the image. Blooming is not an issue for the Legend 515 system because of light accumulation characteristics. Antiblooming is always enabled for the 535, 545, 552C, 554 and 554C systems.

Warning: Do not enable Antiblooming when working with the Measurement Vision Tool when using sub-pixel computations. Results may be affected by as much as one full pixel when Antiblooming is activated.

Illumination

The illumination signal is a digital output, normally configured on pin 8 in a DVT Vision Sensor. Strobe light sources from the DVT product, with part numbers of the form IDxx are designed to use the strobe illumination signal (see Figure 11).



Figure 11: A strobe illumination source (IDRA) that uses the strobe output.

Pulse width for the strobe signal is determined by the Product exposure time (which is a Product-specific parameter). These two controls are tied together so the strobe light remains on for the duration of the image sensor's exposure to light.

For DVT Vision Sensors, this option of illumination enabled/disabled controls the integrated LED ring as well. This will allow the user to gain some control over illumination without using an I/O line. However, when more specific lighting is needed, up to three external strobe lights (besides the integrated LED ring) can be controlled using the digital I/O lines.

Product Identification

Every product in a Vision is assigned a number, called the Product Digital ID, which is used to reference the product. This number enables the user to send a digital signal to the Vision Sensor to indicate which product to use for a specific inspection. In order to do that, a number of I/O lines must be dedicated to the Product ID bit input function. The number of lines dedicated to this function determines the maximum number of products that can be selected externally. Basically, n lines dedicated to Product ID bits will enable the user to select among 2^n products. A common configuration for I/O setup involves designating 4 lines as Product ID bits (bits 0 through 3). The following table reviews the binary combinations for 16 Product ID numbers (binary 0-15) using these 4 input bits:

Table 5. Product ID Bit Binary Patterns

Product ID	Binary Pattern (Bit 3, Bit 2, Bit 1, Bit 0)	Product ID	Binary Pattern (Bit 3, Bit 2, Bit 1, Bit 0)	Product ID	Binary Pattern (Bit 3, Bit 2, Bit 1, Bit 0)
0	0,0,0,0	6	0,1,1,0	12	1,1,0,0
1	0,0,0,1	7	0,1,1,1	13	1,1,0,1
2	0,0,1,0	8	1,0,0,0	14	1,1,1,0
3	0,0,1,1	9	1,0,0,1	15	1,1,1,1
4	0,1,0,0	10	1,0,1,0		
5	0,1,0,1	11	1,0,1,1		

Product Selection

Digitally selecting a Product is a simple process. Refer to Figure 12 for a diagram illustrating the following procedure. First, set the binary bit pattern of a Product's ID number on the "Product ID Input" lines. When the Product ID bits are loaded (in this case Product ID 1, which is 01 in 2-digit binary code) the Product Select line should become active. This tells the Vision Sensor to read the data bits from the Product ID bit lines. When this signal is sent to the system, a response can be read indicating a successful/unsuccessful operation. Two outputs are dedicated to alerting the user of the success or failure of the digital selection process: Product Select Pass and Product Select Fail. Reading one of these should be enough to determine the success of the product selection.

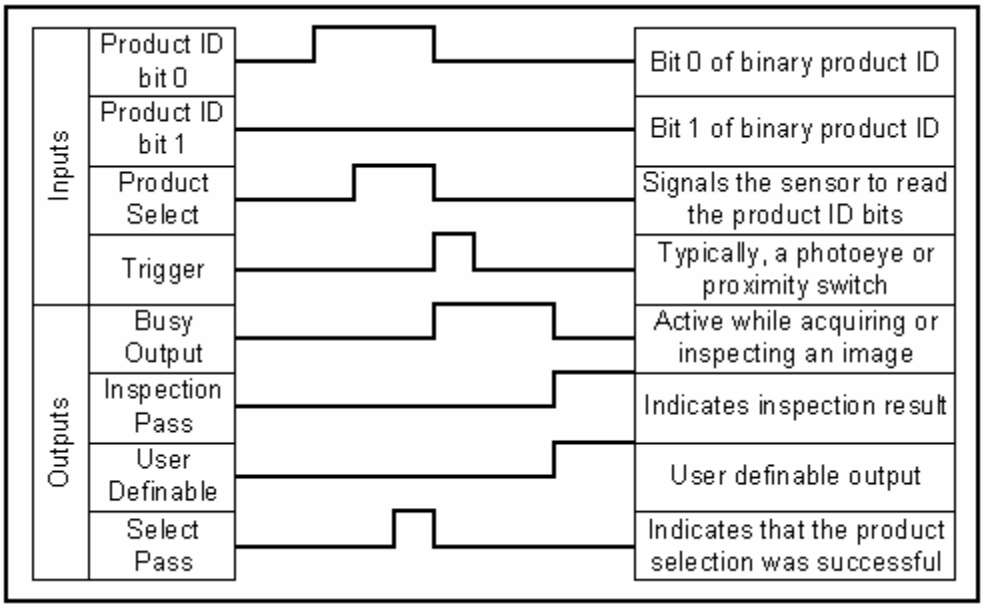


Figure 12: Select a Product using the digital inputs. The above diagram shows the I/O timing of the Product Selection process.

For example if a Product ID of 3 is selected but the sensor has only products with ID's 0, 1 and 2, Product Select Fail should be high and Product Select Pass should be low. When the product selection has been completed the system can be triggered for an inspection.

Product Selection can also be performed over the Ethernet port using a variety of methods. These are explained in the Help Files.

System Graphs

System graphs help users troubleshoot systems as well as gather statistical data about the inspections. The graphs available at the System level are: Inspection Statistics, Inspection Times, and Digital I/O. These features are part of the Intellect user interface; they do not reside in the Vision Sensor.

The inspection Statistics graph is particularly useful for troubleshooting. This graph keeps count of total inspections, inspections that timed out (the inspection did not complete in a certain amount of time), and inspections that could not be taken when trigger was received. The last type of count refers to two cases: resource conflict (the Vision Sensor is triggered while it is acquiring an image) and missed count (the Vision Sensor was triggered while the image buffer was full so the trigger was ignored).These parameters help users determine when there might be a timing issue with the inspections. Figure 13 shows an Inspection Statistics Graph.

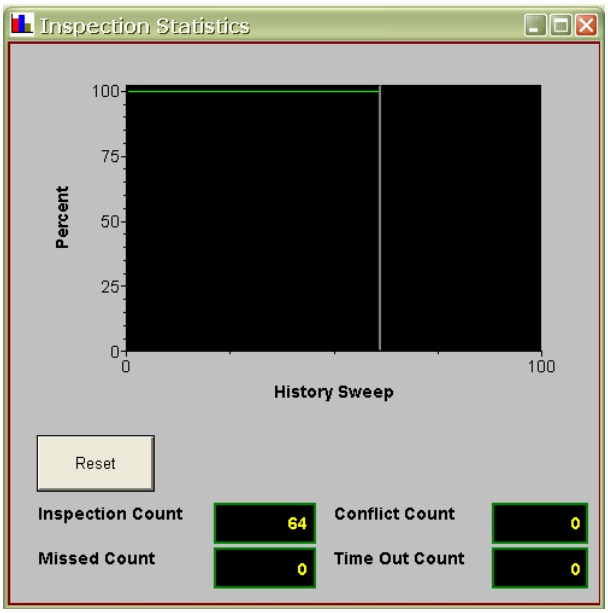


Figure 13: Inspection Statistics Graph

The Inspection Times graph is a very important graph. It shows how much time every inspection takes showing the maximum and minimum inspection times. This graph should be used when two or more different methods for inspections are being tested. Usually, the time it takes to perform an inspection is crucial, so after setting up different Vision Tools to perform a certain inspection, this graph will tell the user which method is the fastest one. Figure 14 shows a screen capture of this type of graph. The information displayed in this graph can also be found in the Result Table and in the Properties Window of the Product.

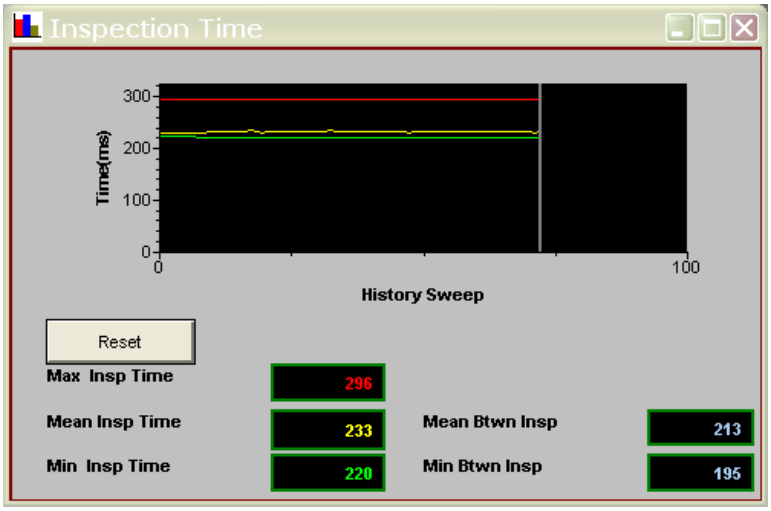


Figure 14: Inspection Times graph.

Finally, the digital I/O graph works like an oscilloscope. It uses time as the abscissa (X axis) and eight configurable I/O lines as the ordinate (Y axis) to show a timing diagram that indicates when signals go active or inactive. This graph is very useful for troubleshooting communications. Figure 15 shows a Digital I/O graph.

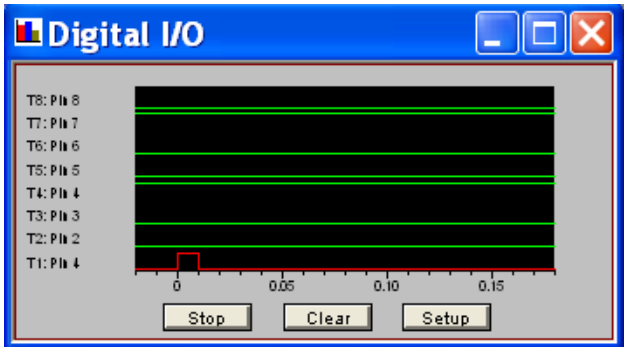


Figure 15: Digital I/O graph. The X axis indicates time in milliseconds.

Product Graphs

Product graphs help users troubleshoot products as well as gather statistical data about the inspections. The graph available at the Product level is the Pass/Fail Graph. The Result Table also displays product information and is available under the View menu. These features are part of the Intellect user interface; they do not reside in the Vision Sensor.

The Pass/Fail graph keeps count of the inspection results. It Shows the actual counts of Fail and Pass, outputs as well as an indication of the percentage of the total inspections taken that those counts represent. The graph shows the number of inspections as the abscissa (X axis) and the running percentages of inspection results as the ordinate (Y axis). Figure 16 shows a Pass/Fail graph.

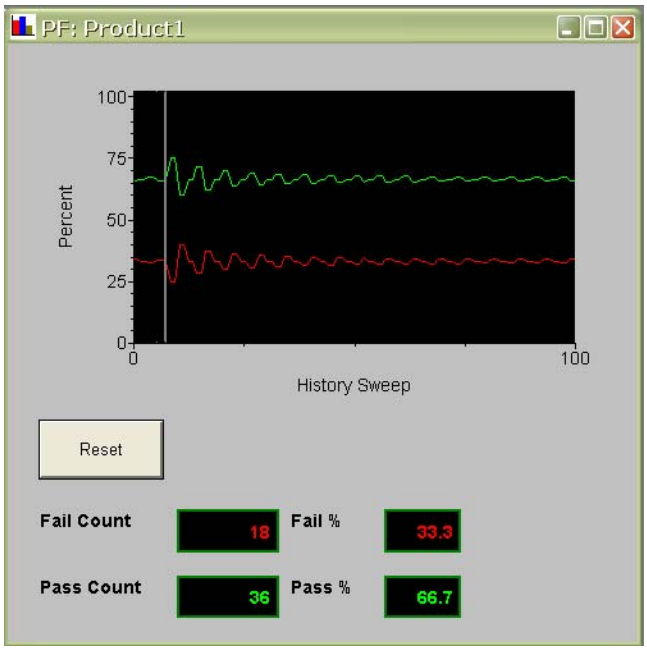


Figure 16: Sample Pass/Fail graph.

The Result Table is a list of all the Vision Tools in the product. It consists of many columns of which the most important are: the name of the Vision Tool, the result for that specific Vision Tool, the Method or type of Vision Tool, the Cause of Failure, the Order in which the tools are processed, and the Timing (ms) information for that tool. Figure 17 shows a sample result table. It indicates the product name, in this case "Scratch Detection", and it shows the Vision Tools being used as well as their outputs.

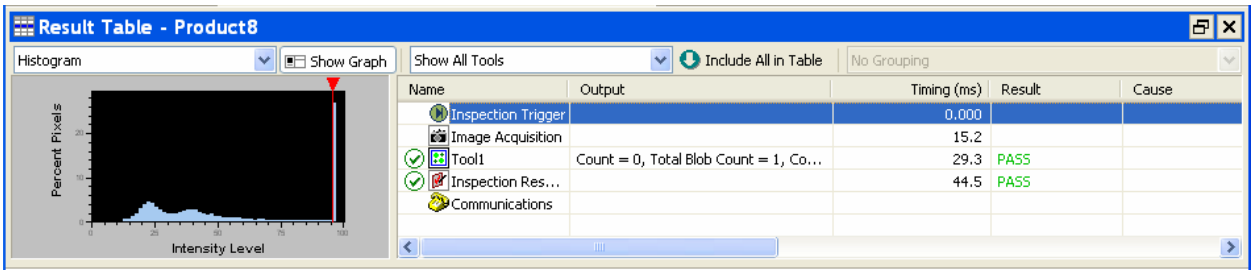


Figure 17: Result Table for a product. It lists all the Vision Tools in it as well as their outputs.

The Inspection Statistics table, accessed by double clicking on Inspection Results in the Result Table, is particularly useful for troubleshooting. This table keeps count of passed inspections, failed inspections, pass and fail percentages, image acquisition timing, inspection timing, inspection overhead, and total time for inspections. These parameters help users determine when there might be a timing issue with the inspections. Figure 18 shows an Inspection Statistics Table.

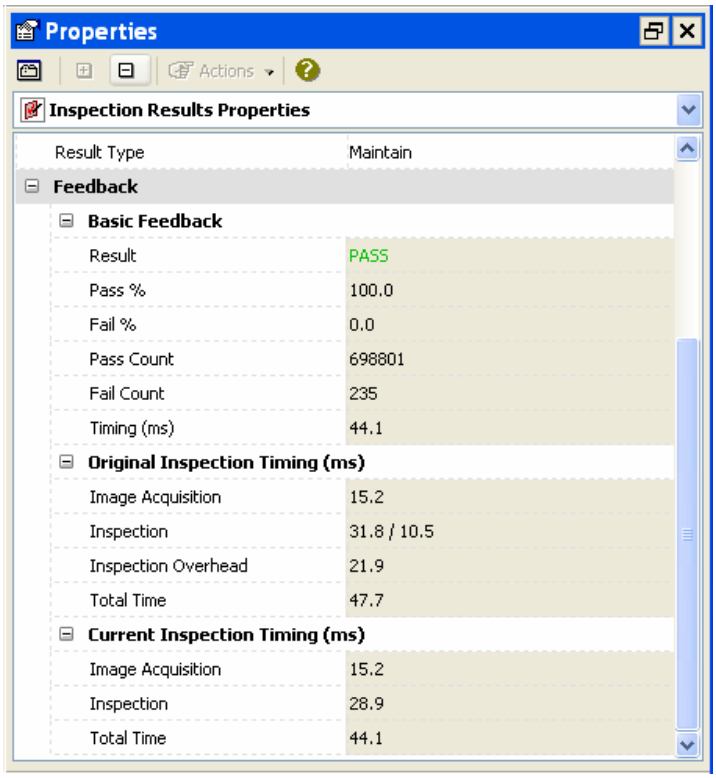


Figure 18: Inspection Statistics graph.

Vision Tool Parameters

Vision Tools are the workers inside Vision Sensors. Each one is assigned a specific task to extract information from the image. The combination of all the tasks performed by the Vision Tools makes up the entire inspection. The main Vision Tool parameters are the Vision Tools shape, processing options , threshold, and PASS/FAIL conditions.

Vision Tool Shape

Vision Tool creation is a very simple point and click process. The first step is to draw a certain shape in the image. The available shapes depend on the type of Vision Tool being used, and the selected shape

depends on the part being inspected. The two main categories of Vision Tool shapes are Line Vision Tools and Area Vision Tools.

Line Vision Tools are mostly used when edges need to be found. These types of Vision Tool scan along lines analyzing every pixel along them.

- Line Segment: Allows the user to draw a line at any desired angle.
- Polyline: Allows the creation of a line defined by straight segments, used for analysis of very complicated shapes.
- Circle: Draws a circle, used for analysis of round or highly symmetric shapes.
- Ellipse: Draws an ellipse, designed for cases where a circle is to be analyzed but perspective errors make circles appear as ellipses.
- Elliptical Arc / Circular Arc: Draws an incomplete circle, used for cases where only part of the ellipse or circular region is to be analyzed.

It should be noted that even though some lines are closed, that is, they draw a figure, the Vision Tools are line Vision Tools and they only analyze the edges of those figures. Only area tools analyze areas.

NOTE: For instructions on how to draw a specific type of Vision Tool select the desired shape and refer to the Intellect status bar for click-by-click instructions.

Area Vision Tools analyze the pixels contained in a specific area of the image. That area is determined when the Vision Tool is drawn.

The different shapes are applicable to different cases; thus the user should be familiar with the application and the part to be inspected and then select a shape accordingly. An important difference between line and area Vision Tools is the processing time. While area Vision Tools might be the best choice to make the inspection reliable, they take more processing time than line Vision Tools. Besides the basic line and area Vision Tools, Intellect has other specific shapes available for certain applications.

Those shapes will be covered as the specific Vision Tools that use them are explained in the next chapter.

PreProcessing Operations

In Intellect Preprocessing is available in the Preprocessing Vision Tool group as well as in individual Vision Tools. Image quality is always the most important factor for setting up an inspection. Sometimes a minor change in the lighting technique makes a big difference for the image. However, in many cases, users have limited control over the physical setup of the system, so another way to deal with noisy images must be used. For those cases, some Vision Tools have a built-in number of operations that process the image before a threshold is applied. These operations do not actually alter the image; they alter the perception of the image for the selected Vision Tool. Since Vision Tools can overlap, altering the actual image would affect the result of every Vision Tool analyzing it. The available operations are:

- Erode: shrinks shapes in the image a user-specified number of pixels. Useful operation when dealing with background noise
- Dilate: expands shapes in the image a user-specified number of pixels. Useful operation when dealing with noise inside the part
- Open: combination of Erode operation followed by Dilate operation. This option eliminates background noise and does not alter the sizes of the parts
- Close: combination of Dilate operation followed by Erode operation. This option eliminates noise inside the part and does not alter the sizes of the parts

Figure 19 illustrates the use of the Erode and the Open operations. In this case, the dark part being analyzed is in presence of background noise.

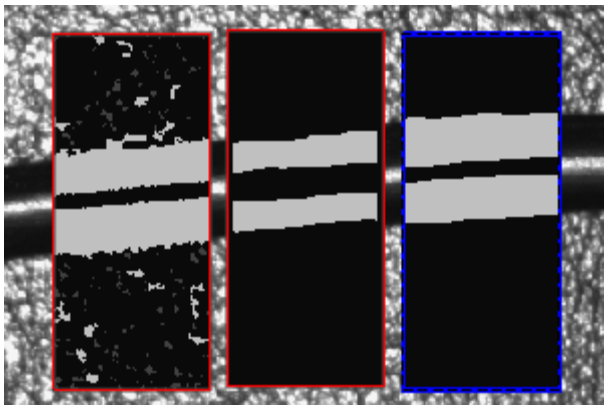


Figure 19: Noisy image illustrates use of Erode and Open operations. The Vision Tool on the left uses neither, the one in the center uses the Erode operation, and the Vision Tool on the right uses the Open operation.

The use of a Vision Tool with no options of image processing would result in an inspection of the part and the noise as shown in the first Vision Tool. The second Vision Tool uses the Erode operation, it shrinks everything (3 pixels in this case) until the noise is gone, but that affects the original size of the part. The third Vision Tool uses the Open operation, that is, after shrinking the parts it expands them. The result is the noise disappearing and only the part to be inspected being expanded to its original size.

Figure 20 shows the inspection of a part with texture. The texture causes some noise inside the part; the first Vision Tool shown does not use any operation, so it picks up all the noise.

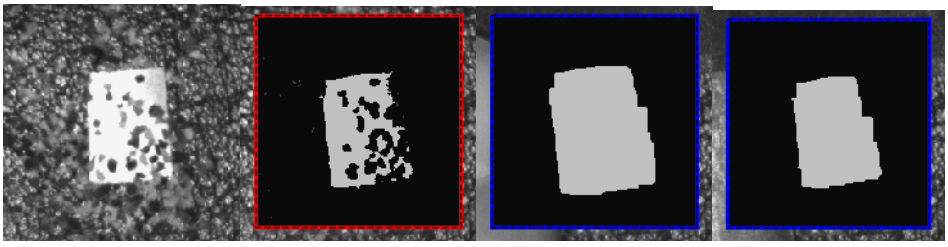


Figure20: Example where the operations of Dilate and Close are used.

The second Vision Tool uses the operation Dilate, which expands the part in all directions. This helps with the noise but alters the area of the part. The last Vision Tool uses the operation of Close and it successfully eliminates the noise and returns the part to its original size.

The user needs to remember that all these operations add some processing time. The operations of Open and Close add twice as much processing time as the other two operations because they perform the basic operations twice.

Threshold

Vision Tools use the information from every pixel to inspect images. Every pixel contains a certain intensity level ranging from 0 to 100. Based on application specifications, the user selects a threshold level to classify the pixels contained in the Tool as dark pixels (below threshold) or light pixels (above threshold). There are several ways to compute a threshold:

- **Intensity Based, Fixed Value:** This option lets the user manually select the threshold level. This threshold has to be a number between 0 and 100 and it is considered a static threshold because it does not change from image to image.
- **Intensity Based, Percent of Path Contrast**: This is a dynamic threshold that places the threshold at some location between the minimum and maximum intensity values based on the

value set by the user. This threshold is calculated as follows: $Threshold = (I_{MAX} - I_{MIN}) \times T\% + I_{MIN}$. The "*T%*" represents the threshold percent set by the user, I_{MAX} and I_{MIN} are the maximum and minimum intensity values inside the area that the Vision Tool is set to analyze. This option selects a threshold which can adapt itself from image to image (dynamic threshold) which makes it more robust that any static threshold.

- **Intensity Based, Local Difference Method (for 1D Reader):** The local difference method uses a dynamic threshold based on a small sample window. This method is used for uneven lighting or very small (less than three pixel) transition between light and dark. This method evaluated the surrounding minimum and maximum intensities to establish an edge. Used for 1 Reader Vision Tools.
- **Intensity Based, Auto Bimodal:** This method automatically calculates a single threshold value to use based on the entire histogram. Since it uses all the values in the area to calculate the threshold, the effects of noise and specular reflections are minimized. Auto bimodal thresholding is recommended when there is a distinct foreground and background intensity. The threshold is recalculated for each image.
- **Intensity Based, Adaptive (for OCR Reader):** The system calculates a specific threshold value for different areas of the image depending on the contrast found in those areas. The user needs to specify a window size in pixels, the software then calculates a threshold (using the percent of path contrast method) for individual windows, making the overall threshold more robust when lighting and contrast are not uniform. This method is the slowest of the thresholding methods.
- **Gradient Based, Fixed Value:** This method for calculation of a threshold uses the pixels intensity gradient. That is, it compares pixels with their neighbors in the image to determine local changes in intensity. It then produces a gradient graph that has peaks where the highest changes are located. The user needs to specify a threshold to separate the main edge(s). This method works in the same way as for Intensity based threshold (as explained above). **NOTE: This method uses pixel interpolation to calculate the best position for the edge, providing sub-pixel accuracy.**
- **OCR specific:** the OCR Vision Tool has three extra options for calculation of the threshold that will be discussed later on when the OCR Vision Tool is covered.

Vision Tool Result

Vision Tools can be set to PASS or FAIL based on predefined criteria. Usually, the PASS condition is assigned when the parameter being checked is within the specifications. The FAIL output is reserved for cases where the parameter being checked exceeds the specifications. Every Vision Tool contains a custom set of rules for the user to set as the PASS/FAIL criteria. Those rules are related to the specific functionality of the Vision Tool. That is, for measurement Vision Tools the condition would be based on maximum and minimum size, for the barcode reader it would be set based on a matching code, etc. However, there is an option that is common to most Vision Tools: Minimum Contrast. The option of Minimum contrast should be used when a dynamic type of threshold is being used. The use of this option will be illustrated with an example: suppose the user selects Percent of Path Contrast as the method for calculation of a threshold. The Vision Tool (as explained before) will determine the intensity levels of the brightest and the darkest pixel, and based on those numbers it will compute a threshold level. This is a very powerful method, but what happens when the part is not present? Only the background is present, so the Vision Tool sees a very low contrast given by the noise in the background. Based on this contrast the Vision Tool still computes a threshold for the image. Figure 21 shows the pixel graph for a Vision Tool illustrating this case.

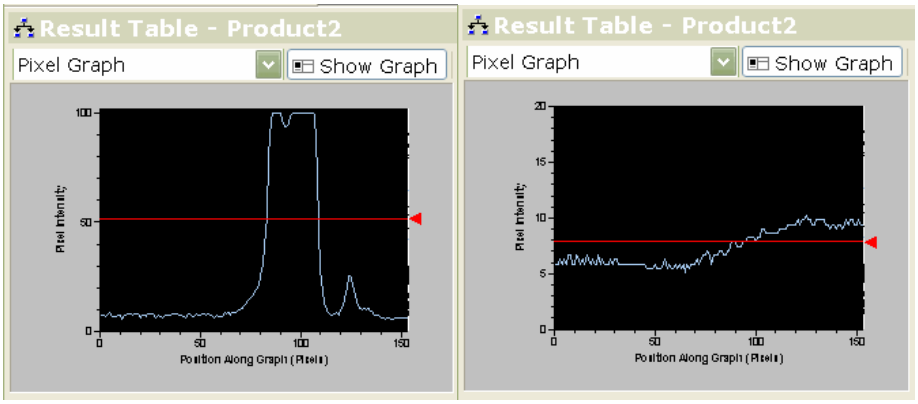


Figure 21: Case where the part being inspected (determined by the group of light pixels in the graph on the left) is not present in the next image. The graph on the right shows that there is still a threshold being calculated.

The pixel graph on the left shows the presence of the (light) part. The threshold is correctly calculated at 50% of the difference between the lightest and the darkest pixel. The pixel graph on the right shows the pixel graph for the same Vision Tool for the case where only background is present in the image with intensity levels between 5 and 12. In this case, there is no part, but the Vision Tool still computes a threshold because it can still find a lightest and a darkest pixel. In order to avoid this case, the user can select the "Minimum Contrast" option and set a minimum contrast that must be matched to even compute a threshold. For the second case explained above, a minimum contrast of about 10 would prevent this Vision Tool from computing a threshold and thus will cause it to fail.

Vision Tool Graphs

Just like at the Product level, Intellect offers graphs to display Vision Tool-level information. This information is tool-specific; it could be generic information (Pass/Fail, etc.) or very specific Vision Tool information. The latter depends on the Vision Tool being used. The first and most common Tool graph is the Pass/Fail graph. It serves the same purpose as its product-level counterpart, but the data refers to a specific Vision Tool (see Product Graphs for more information). The rest of the Vision Tool graphs could be classified as Image Graphs.

Image graphs are graphical interpretation of the areas analyzed by the Vision Tools that help the user set a threshold. There are three types of image graphs: Pixel Graph, Histogram and Gradient Graph. The Pixel Graph is the simplest way of displaying information about a Vision Tool path. It uses the Vision Tool path as the x-axis and the intensity of the pixels as the y axis. That is, the range displayed in the X axis goes from the origin to the end of the line Vision Tool, whereas the range in the Y axis goes from zero intensity (black) to 100 (white). Figure 22 shows the pixel graph for a Vision Tool with a length of 408 pixels.

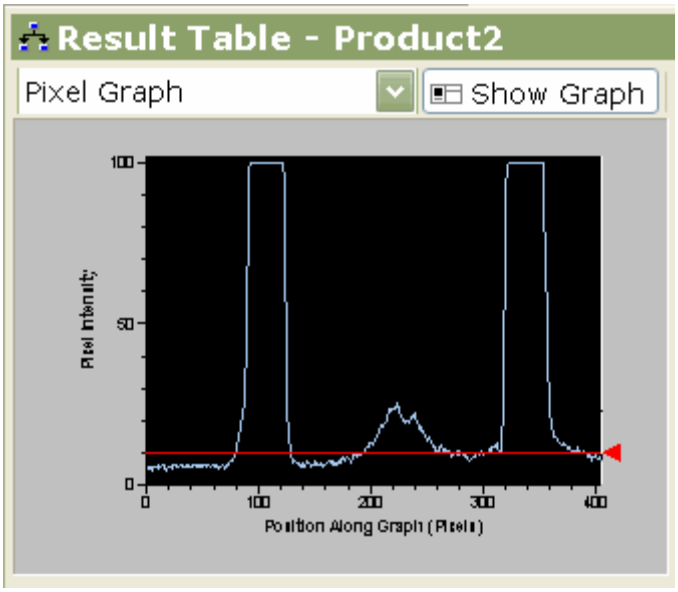


Figure 22: Example of a Pixel Graph for a line Vision Tool of 408 pixels in length.

As the graph illustrates, the pixels near the center of the Vision Tool are dark (low intensities), then, a light region is found so the graph peaks up to the high intensities and then it goes back down towards the low intensities as it approaches the end of the Vision Tool. This occurs on each side of the center.

Histograms are also used to display the intensity of the pixels, but in bar chart format. Unlike pixel graphs, histograms use the intensity levels (ranging from 0 to 100) as the abscissa; for the ordinate, they use the percent pixels. That is, histograms display the percentage of the pixels found at every intensity level. Figure 23 shows a histogram for an intensity Vision Tool. The histogram shows two well defined groups of pixels which may represent background (darker pixels) and the actual part (lighter pixels).

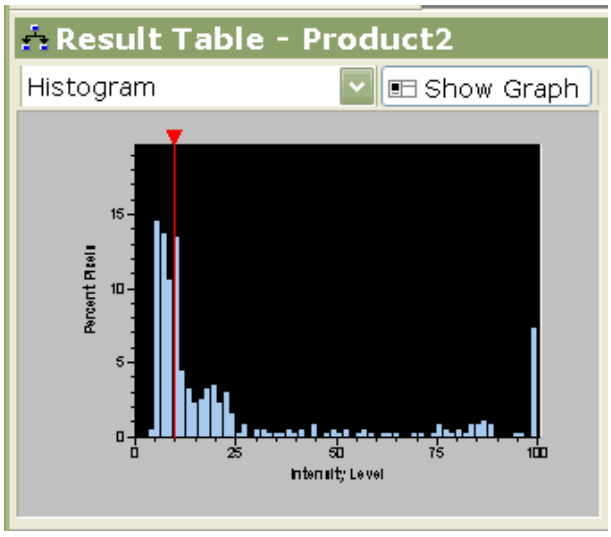


Figure 23: Screen capture of a histogram.

The third image graph is the gradient graph. This type of graph plots changes in intensity values from pixel to pixel along the path of the Vision Tool. It does not show the actual intensity value of the pixels,

just the changes. Sharp positive or negative peaks represent strong edges. Figure 24 shows a pixel graph and a gradient graph for a certain Vision Tool.

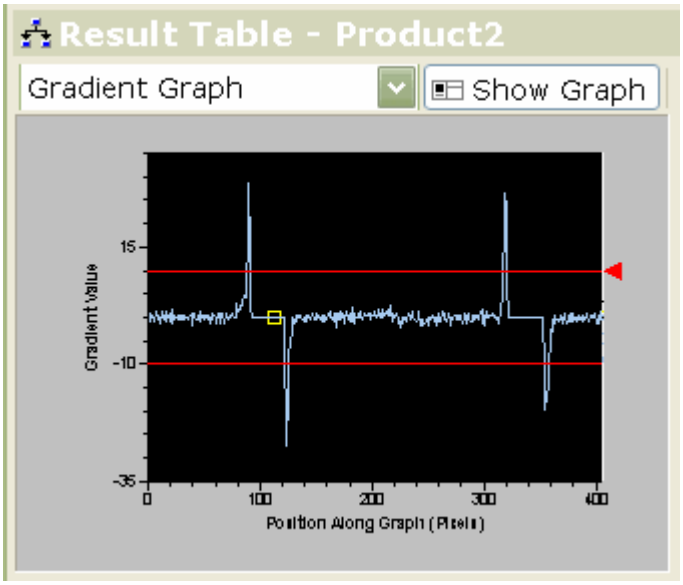


Figure 24: Gradient graph for a Vision Tool.

Notice how the gradient graph peaks where the pixel graph shows the biggest change in intensity levels.

Digital Relearn

Digital relearn is an option that is present in many Vision Tools. It refers to cases where the inspection consists of comparing something to a learned version of it. Its use extends to the following Vision Tools: Area Based Positioning, Pixel Counting, Area Based Counting, OCR, 1D and 2D Readers, Label Inspection, Bead Inspection, Identification, and Flaw Detection. When these Vision Tools are created, something is learned (a shape, a code, a color, etc.). When the Vision Tool performs inspections, it uses the learned model. If the application happens to change the elements being used, there is no need for the user to connect to the Vision Sensor using Intellect and reconfigure the Vision Tool. A simple digital signal to the Vision Sensor can be used to indicate that a new element is ready to be learned. In that case, the system takes the element (color, code, shape, etc.) from the next image to be inspected and stores it as the new model.

Intellect Software

Now that the user is familiar with the system, product and tool level parameters, we will discuss how to access them. This section explains the basics of the Intellect software. After reading this section the user should be able to access all the parameters explained before and should be more comfortable with the system. Intellect consists of two main parts. One of them is the Intellect firmware which resides in the Vision Sensor. The Intellect firmware is responsible for all the computations that happen inside Vision Sensors. The user has direct control over the firmware by programing the user interface, which translates everything for the firmware to execute. The second part of Intellect is the software. The Intellect software consists of the Intellect user interface and the Vision Sensor hardware emulator. The user can only access the components that reside in the PC: the Intellect user interface and the hardware emulator. Using the Intellect user interface the user connects to a Vision Sensor and makes changes to the product or system files loaded in the Vision Sensor. In order to do this, the Intellect user interface queries the Vision Sensor for images and reproduces in the PC what the Vision Sensor does in its internal memory with those images.

Intellect User Interface

The Intellect User Interface is a Microsoft Windows® application used for the configuration of Vision Tools. The main areas of the interface are the main menu, the toolbars, the Vision Tool toolbox, the Video Display, the result table, the properties window and the status bar. Figure 25 shows a screen capture of the Intellect user interface highlighting these areas.

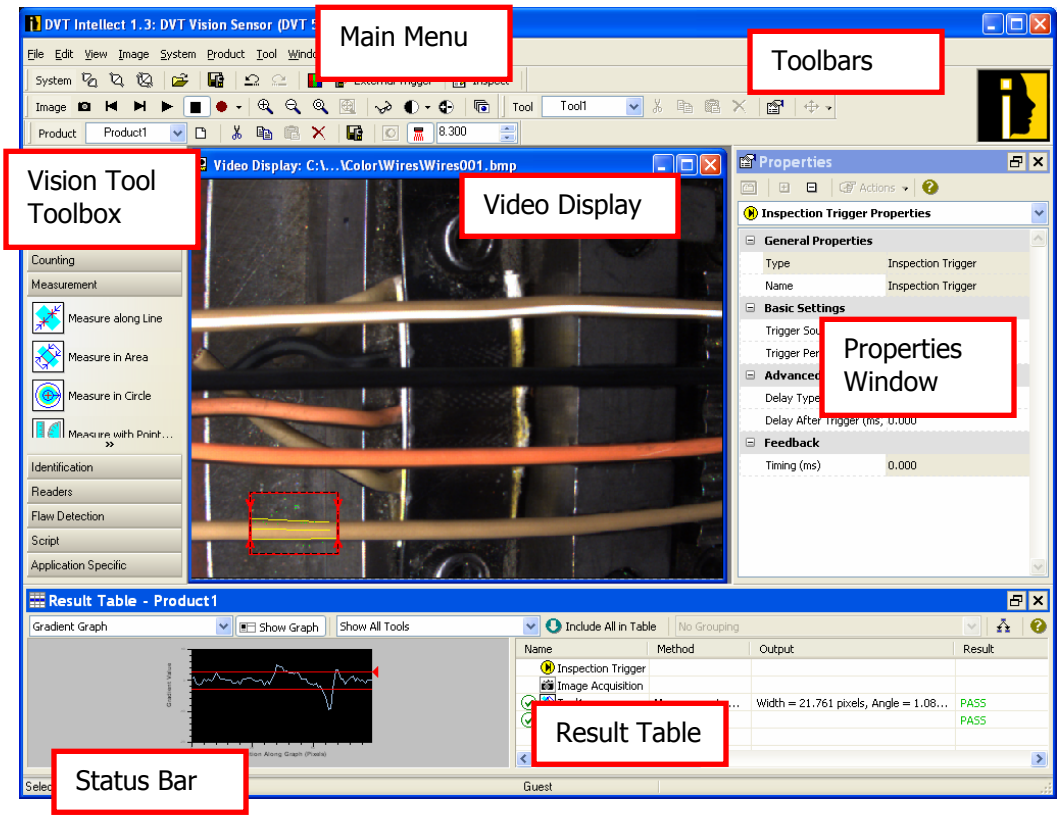


Figure 25: Screen capture of the user interface highlighting the most important areas.

The Main Menu contains commands to control most of the parameters discussed so far. Here, we will discuss the functionality of each submenu:

- The "File" submenu contains commands to open and save product files, system files, and graph information. The file menu also contains log in as well as manage access control and log events.
- The "Edit" submenu is used to Copy, Cut and Paste Vision Tools and Products. In addition, the edit menu contains the undo command.
- The "View" menu contains all the different view windows that can be opened in the user interface. This includes the Vision Tool Toolbox window, the Properties window, the Script Debug window, the Result Table as well as others. If one of these windows is closed, selecting it under the view menu make it appear again.
    - Terminal Window: Opens an Ethernet connection to the Vision Sensor's system terminal (port 3246 for new systems and 5000 for old systems). The DVT Vision Sensor can receive a number of commands in this port that give the user access to it without using Intellect. By using this terminal window, the user can execute these functions on the Vision Sensor without exiting Intellect. Refer to the Intellect help files for a complete list of commands Vision Sensors can receive on the system terminal.
    - Script Debug Window: Special window dedicated to receiving Debug statements from scripts that help in troubleshooting the code. Please refer to the script documentation for more information about it.
    - Datalink Output Window: The DataLink View window shows the information that is sent out the data after each inspection. The information that is sent out the datalink port is configured for each product in the Communications Manager.
    - Options: Gives access to advanced features. From this menu the user can enable advanced features for many Vision Tools, set parameters for the Vision Sensor itself, and activate/deactivate buttons from the Vision Tool toolbar.
- The "Image" menu gives the user control over many imaging parameters. Most options in this menu are very general such as play, stop, record, create static image, zoom in, zoom out, etc. The image menu also allows you to configure the image sequence when connected to an emulator.
- The "System" menu contains managers and settings that are at the system level including background scripts, communications, network connections, calibration, coordinate transformations, and system administrations.
    - Network Explorer: This options opens the network communications managers which allows you to connect up to DVT systems.
    - System Explorer Window: This window allows easy and organized access to the following managers and settings:
        - Background Scripts Window: From this window, the user can add, delete, edit, and create background scripts. Please refer to the script documentation for information about background scripts.
        - Coordinate Systems: Opens the system manager to the coordinate system section which allows coordinate transformations to be defined. This feature is most often used with robotic and measurement applications.
        - Product Manager: Opens a listing of all products in the system where the user can rename the products, specify the Power On product, the Inspect Product, and the Selected Product. The user can also apply coordinate transforms in this window.
        - Retained Images: The retained images option allows past failed images to be viewed.

- Communications Settings: This option opens the communications manager which is used to set up communications between the Vision Sensor and other industrial devices. Communications include digital IO, Ethernet TCP/IP, Modbus, etc.

  - System Calibration: The calibration dialog allows the Vision Sensor to be calibrated for Sensor Gain, FOV balance, White Balance, and Reflectance.
  - System Administration: The system administration section contains information on flash memory, security, users, and licenses. It also contains information about image buffers. The size of the image buffers is often changed when using Line Scan systems.

- The "Products" menu contains product level information and settings. This includes product level communications settings like datalink strings, SmartLink data transmissions, image acquisition properties, and product management functions.
- The "Tool" submenu offers access to Vision Tool parameters. From here the user can edit Vision Tool settings and create Vision Tools.
- The last two options, "Window" and "Help", have standard Windows® functionality. The "Window" option is used to customize the user interface. The "Help" menu brings up the Intellect help files, which are rich in content and examples.

The Toolbars contains buttons that offer shortcuts to the most frequently used functions of the software. Every button offers a tool tip, that is, when the cursor is placed over a button, a message is displayed on the screen indicating the functionality of the button. The toolbar area contains four toolbars; System, Image, Product and Tool toolbars. The System toolbar contains buttons to connect/disconnect from the system, open a file, undo an operation, white balance, save system to flash, and settings for inspection triggering. The Image toolbar contains buttons to change the exposure time (which is displayed in milliseconds), a button to control the strobe lights, antiblooming control, and controls for both the Vision Sensor and the user interface. The controls for the user interface contain a "Play" button, a "Stop" button, and a "Record" button. The "Play" Button is for the user interface. The fact that the user interface gets images from the Vision Sensor does not mean that the Vision Sensor is inspecting them. The only indication that we have of that is the availability of the inspection results from the Vision Sensor, and the control the user has over this process is the "Run/Stop Inspections" button. If this is enabled, the Vision Sensor is inspecting images, which means that it is making the output signals available. Table 6 summarizes the different cases.

Table 6. Effects of User Interface Play Mode

User Interface: Play Mode	Inspection Mode	Real Time Feedback to user interface	Outputs from the Vision Sensor
PLAY	OFF	Active (updates images)	Not Available
STOP	OFF	Inactive (stopped)	Not Available
PLAY	ON	Active (updates images)	Available
STOP	ON	Inactive (stopped)	Available

The user should understand the differences between these two options in order to avoid incorrect configurations. The main reason for the independence between these two functions is that the Vision Sensor is a stand alone System. It does not need a user interface connected to it in order to inspect images, so it must not rely on user interface modes to produce inspection results. Likewise, the user does not have to rely on the Vision Sensor running inspections to see images, and it does not have to stop the Vision Sensor from inspecting to observe a particular image. For example, if the inspection mode was tied to the user interface play mode, the Vision Sensor would stop inspecting images every time the user needs to analyze an image in details for which it would be necessary to stop the real time feedback. This

would cause a problem when other devices depend on the result from the Vision Sensor to accept or reject a part. It should be mentioned that Table 6 summarizes what happens upon triggering the system. That is, when the trigger mode is set to internal with a period, the updating of images/outputs as set by the aforementioned parameters will occur periodically. However, if the trigger mode is set to external, the updating of those parameters will only occur when the Vision Sensor receives an external trigger.

In addition to these controls, the Image Toolbar contains a Previous Image and a Next Image button. The Previous Image button works with the Static Image Window and the Sample Image Display. It changes the current image shown to the previous image in the sequence being used.

The Next Image button acts as an external trigger button. When working with an Internal Trigger the next image button changes the current image shown to the next image according to the image sequence when the Play button is not pressed. If the system is on external trigger this button is enabled and its functionality affects the Video Display (thus the Emulator or Vision Sensor) by causing it to go to the next image. For the Emulator it means that the next image in the selected sequence is displayed. For the Vision Sensor it will trigger the system to obtain a new image.

In order to determine the function of the Next Image button that is used, users need to select the Static Window (for the first function) or the Video Display (for the second function) by clicking once on it before clicking on the button.

The Product toolbar and the Tool toolbar contain similar functionality for products and Tools. For products it offers the extra options of showing the result panel (explained below), creating a new product, and saving a product to flash memory. Flash memory is the nonvolatile memory where the product should reside in order to remain in the Vision Sensor after the power is cycled. Both sections of the toolbar contain a pull down menu with the existing products in the system and the Vision Tools that the selected product contains. The remaining toolbar buttons are to duplicate, edit parameters, delete, or access the graphs associated with the product/Vision Tool selected from the pull down menu.

The Video Display, is where the images obtained from the Vision Sensor are displayed. Not every image from the Vision Sensor is transferred to the user interface. For the Vision Sensor the number one priority is to run inspections. After that, and depending on processor usage, it sends images via Ethernet to the user interface. In cases where the Vision Sensor is being triggered very fast, the user interface will only receive some images. This is why the display is called sampled image display; it only displays sample images from the Vision Sensor. Besides the case of the Vision Sensor being busy with inspections, other factors influence the number of images transferred to the user interface such as network traffic, the types of hub being used, the Ethernet card being used, the speed of the processor in the PC, and the type of images being transferred (color, high resolution, etc.). The top bar of the Video Display shows a number called the Image ID, which is an incremental number assigned to every image. The Image ID is reset on power-up. This number should only be used to monitor the rate at which images are being transferred.

The Vision Tool toolbox is where all Vision Tools are available. This toolbar represents a shortcut to most options available under the Vision Tool menu of Intellect. If the user prefers to use the main menu, the toolbar can be hidden to maximize the work area by deselecting the "Toolbox" option from the View menu or pressing the close button at the corner of the Toolbox.

The Result Panel is a combination of two elements that have been discussed in this chapter. The result table and the Tool graph for the Vision Tool that is selected from the result table. When a Vision Tool has focus in the Video Display, the too will be highlighted in the result table and the Tool graph to the left of the result table with correspond to this Vision Tool.

The Status Bar, the last section of the user interface contains three main sections:

- Instructions section: This is the left side of the bar. When the user selects to create a new Vision Tool, it displays click-by-click instructions so the user does not need to remember how to draw every Vision Tool.

- User section: This section is related to the Vision Sensor administrator, it shows the username under which the current operator is logged on. It will show "Guest" when no user has logged on.
- Image information section: This part of the toolbar is used in combination with mouse movements. When the user moves the pointer over the image being inspected, this section of the status bar will show three values for grayscale systems and five values for color systems. The first two values in both cases are the X and Y coordinates of the pixel where the pointer is located; "X" coordinates start at zero on the left edge of the image and increase to the right, and "Y" coordinates start at zero on the top edge of the image and increase downwards. This section also shows calibrated coordinate system values if available. For grayscale systems there is a third value which indicates the intensity of the pixel (between 0 and 100). For color systems there are three values instead of an overall intensity value, these numbers represent the Red, Green, and Blue intensity values of the pixel.

DVT Vision Sensor Hardware Emulator

The DVT Vision Sensor Hardware Emulator acts like a Vision Sensor, but it is a software application that runs on your PC. When the emulator application is running, the Intellect user interface can connect to the emulator – just like a connection to a Vision Sensor. Actually, Intellect does not recognize a difference between the actual Vision Sensor and the Emulator. The main uses for the Emulator are:

- **Education / Training:** With only a PC, users can run the emulator application with Intellect and practice using the tools to learn how Intellect operates.
- **Off-line Setup:** Take development of Product files off of the factory floor. Use the emulator to continue setup work on Products even after you've walked away from the machine.
- **Troubleshoot Reliability Problems:** Is your Vision Sensor rejecting too many good parts or passing flawed parts? Save a batch of the images and work off-line to find the problem, and then perform the necessary changes to improve the inspection and reload the product file to the Vision Sensor.
- **Demonstrations:** Show the power of Intellect without the Vision Sensor hardware present.

Intellect's functionality is opened up only after a connection is established to a Vision Sensor or to the Hardware Emulator. When the application starts, the Network Explorer Manager is opened. Users may opt to connect to a Vision Sensor (hardware device) via Ethernet or to the Emulator. Under Emulator, the user will find a number of options. Each one corresponds to a different model of Vision Sensor. The selection of emulator depends on the Vision Sensor that the user needs. When the user selects to connect to an Emulator, a new Windows® application launches in the background. The basic steps to work with the Emulator are:

- Create a backup copy of the system file or product file that resides in the Vision Sensor. This step is needed only if the user needs to work on the setup of a Vision Sensor. A product file contains all the product parameters (specific for every inspection) whereas the system file contains all the parameters in the System.
- Record some images from the Vision Sensor. Using the record option, the user can setup Intellect to record images imported from the Vision Sensor to the computer hard drive. Then, when Intellect connects to the Emulator, the image sequence can be loaded into the Emulator and the user can cycle through those images. (This step is optional, as the user may already have an image sequence to work with.)
- Start Intellect and connect to the Emulator. Choose the Emulator that corresponds to the Vision Sensor used.
- Perform changes to the system/product files as needed. The Emulator supports any type of changes: the user can add or delete Vision Tools and products, set up communications, etc. The

only changes that are not reflected are changes that relate to physical setup such as exposure time, illumination, etc. because the Emulator works with prerecorded images.

- Back up the product/system file to the PC and restore it to the Vision Sensor.

For a quick tutorial on this process see Appendix A – Emulator tutorial.

Chapter 3 - Vision Tool Reference

This chapter explains the main concepts of the Intellect Vision Tools. As it was mentioned before, Vision Tools are the workers inside Vision Sensors. Every type of Vision Tool serves a specific purpose, and the combination of the Vision Tool results represents the overall result of the inspection. The main groups of Vision Tools are Filters, Positioning, Counting, Measurement, Identification, Reader, Flaw Detection, Script, and Application Specific Vision Tools. Positioning Vision Tools help locate parts to be inspected. In most applications, this type of Vision Tool becomes the key to a successful setup because they locate the part to be inspected and pass an internal reference to the remaining Vision Tools indicating the location of the part. Counting Vision Tools are often used for Presence/Absence applications were the application is to detect features. Specific Inspection Vision Tools include 1D and 2D code readers, OCR readers, measurement Tools, math tools, color Tools, identification, flaw detection, spectrograph for color analysis, and programmable Tools.

Intellect Vision Tools

This section contains information on the different Vision Tools contained in Intellect. These include

Preprocessing Vision Tools

The Preprocessing Vision Tool category contains Vision Tools that alter the image to help the other inspection Vision Tool accomplish their tasks more effectively. In general, the preprocessing Vision Tools reduce background noise, increase edge or foreground contrast, or move image pixels around.

Positioning Vision Tools

The Positioning Vision Tools are a group of Vision Tools that are generally used to find the location of a part or a feature of a part. The Vision Tools can then use the position information to either move other Vision Tools to the proper location for inspection or to guide an external robot or motion controller to the location.

Counting Vision Tools

The Counting Vision Tools are a group of Vision Tools that are generally used to find a number of features of a part or the objects themselves in a region. This information can be used to pass or fail an inspection for presence/absence, orientation, or count.

Measurement Vision Tools

The Measurement Vision Tools provide accurate measurements of objects and parts. This group also contains Datum and Statistics tools.

Identification Vision Tools

The Identification Vision Tools are used to recognize or identify a particular part or feature from a set of learned parts or features.

Reader Vision Tools

The Reader Vision Tool Category is comprised of 1D Barcode readers, 2D Barcode readers, and OCR and OCV Vision Tools.

Flaw Detection Vision Tools

The Flaw Detection Vision Tool Category provides Vision Tools which find subtle flaws in parts.

DVT Script Tool

DVT Script uses a DVT scripting language to expand the capabilities of the Image Inspection System.

Application Specific Vision Tools

The Application Specific Vision Tools are customized for very specific inspection tasks.

Counting Vision Tools for Presence/Absence

The Counting Vision Tools are generally used to simple Presence/Absence applications and counting applications. These Vision Tools perform basic checks for detection of features and edges or analyze the intensity level of the pixels in a certain area or linear path. These Vision Tools are:

- Count along Line Vision Tool
    - Edge Counting: These Vision Tools look for and count transitions from bright pixels to dark pixels and vice versa.
    - Feature Counting: These Vision Tools are an enhanced version of the EdgeCount Vision Tools: besides detecting intensity transitions, they also report the size of the regions between such transitions.
    - Pattern Counting: This Vision Tool learns a pattern and counts the number of instances of that pattern in the image.
- Count in Area Vision Tool
    - Blobs: This Vision Tool counts the number of blobs found in an image, based on user specified blob parameters.
    - Object Locate: This Vision Tool finds the number of instances of a part or feature based on the learn region and user specified parameters.
    - Pattern Match: This Vision Tool finds the number of instances of a pattern in an image.
- Pixel Counting Vision Tool: These Vision Tools count the number of pixels in a given region of the image that meet an intensity threshold or color criteria.

With the exception of Object Locate and Pattern Matching, the classification of pixels as "bright" or "dark" is based on a user-defined threshold. For details refer to section "Threshold" on page 41

EdgeCounting: Count along a Line Vision Tools

The Count along a Line Vision Tool in Edge Counting mode is designed to detect and count transitions between bright and dark pixels (or above and below a threshold). The Vision Tool basically analyzes the image and determines where the pixel graph intersects the threshold line. Those intersections determine an edge, and the user can select the type of edge to locate (bright to dark, dark to bright, or either edge). Figure 26 shows the Count along a Line Vision Tool and its pixel graph.

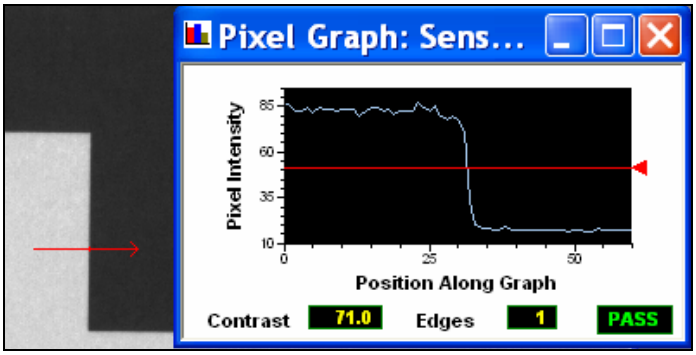


Figure 26: Count along a Line Vision Tool and its pixel graph. The Vision Tool starts on the bright area (high intensity) and then goes into the dark area (low intensity).

In this case, the shape of the Vision Tool is a horizontal line, and the direction of scanning is from left to right as indicated by the arrowhead at the right end of the Vision Tool. From the pixel graph the user can

determine that the length of the Vision Tool is about 60 pixels and that the edge (transition from bright to dark) is located close to pixel number 32. The output of this type of Vision Tool is the number of edges found; in this case it is one. However, the user must be careful to select the correct type of edge to be located. If in this case the user had selected dark to bright edges only, the Tool would have returned zero edges because there are no such edges in the direction of scanning.

Count along a Line Vision Tools are available only as line Vision Tools because of their purpose. Since they detect edges, they must consist of lines searching for transitions between dark and bright pixels along a certain path. As a consequence, a single dark pixel (like background noise) could make them find extra edges and thus alter the result of the inspection.

This type of Vision Tool can be set to pass or fail based on maximum edge count, minimum edge count, finding the appropriate number of edges, and minimum contrast to find edges.

FeatureCounting: Count along a Line Vision Tools

The Count along a Line Vision Tool in Feature Counting mode is designed to detect and count features that make the part being inspected different from the background. A feature, as interpreted by Intellect, is defined by two consecutive opposite edges. For example, a transition from bright to dark, followed by a transition from dark to bright indicates the presence of a dark object in bright background. That object is considered a dark feature. Likewise, a light feature is defined by a transition from dark to bright followed by a transition from bright to dark. The Vision Tools basically analyze the image and determine where the pixel graph intersects the threshold line (edges), then, group those edges to define features. Figure 27 shows a FeatureCount Vision Tool and its pixel graph. It can be observed that the Tool crosses four dark regions and five light regions.

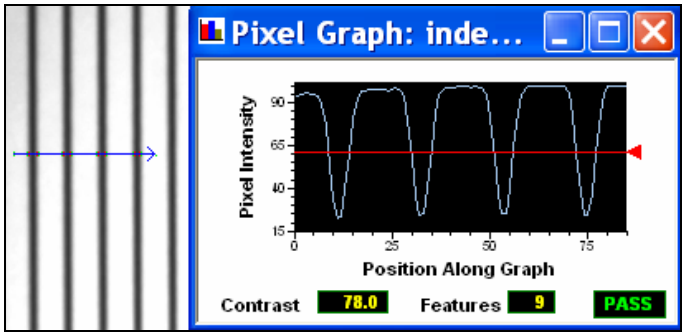


Figure27: FeatureCount Vision Tool and its pixel graph.

This is reflected in the pixel graph, where four groups of pixels can be identified below threshold and five above threshold. These groups of pixels represent the features. As the pixel graph shows, this Vision Tool is finding nine features (four dark features and five light features). Depending on the application, different approaches can be taken. The user can select which features to count. If the user selected to count only the dark features, the Tool would report four features instead of nine. Likewise, if light features were selected, five features would be reported. The default option (used in this example) is to count both light and dark features. If the image to be inspected is not as good as the one in Figure 27, the user may use some options that will make the Vision Tool less susceptible to noise.

The first option to handle noise is to expand the Scan Line Width, which works the same way as in Edge Counting mode (explained before). The second option is a new capability that is unique to Vision Tools that find/count features. It consists of setting a minimum and maximum feature sizes and the noise filter. When this option is used, the only features that are counted are the ones whose size falls within a specified range. The Noise Size filter is use to disregard small noise pixels. Figure 28 shows a sample image with a FeatureCount Vision Tool in it and a pixel graph. The Vision Tool is trying to detect the presence of the dark mark but the dark dot on the outside is another feature. According to the pixel graph, more than one dark feature should be found. However, it can also be observed that whereas the

size of the feature to be located is about 80 pixels, the rest of the features do not exceed 20 pixels in size. The mouse can be hovered over the different features on the graph to get their individual lengths.

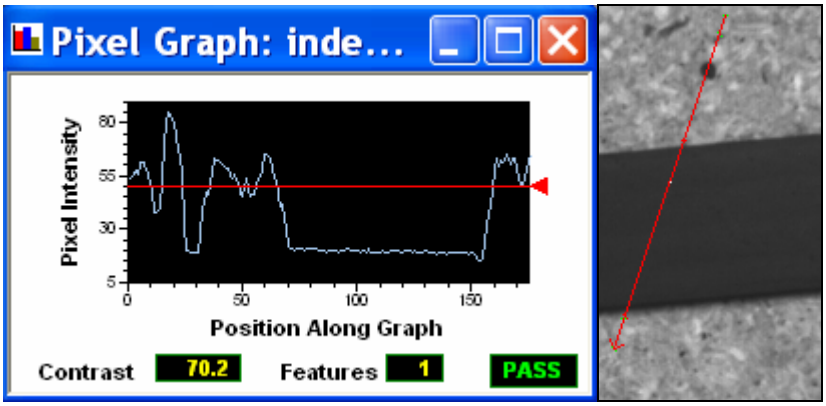


Figure28: Filtration of noise using feature size.

In this case the user can select a minimum feature size of fifty pixels and only one dark feature will be reported by the Tool. This makes the Vision Tool less susceptible to noise making the inspection more reliable.

The Vision Tool in this mode can be set to pass or fail based on maximum feature count, minimum feature count, and minimum contrast to find features.

PatternCounting: Count along a Line Vision Tool

The Count along a Line Vision Tool in Pattern Counting mode is designed to detect and count features based on a learned pattern of grayscale pixels. The mode can be very powerful at filtering out noise if the part features is consistent in it pixel intensity and pattern. The Vision Tool uses normalized grayscale correlation to find the closed matched pattern along the line that matches the features trained pattern. A pattern is a sequence of pixel intensities. Since the Vision Tool learns the grayscale pixel intensity values, no threshold is used for finding the pattern. It is best used when detecting only edges or features does not provide for an adequate inspection and when the part has a signature of pixel values.

Pixel Counting Vision Tools: Intensity and Color Counting

The Pixel Counting Vision Tool calculates the amount of pixels within a threshold range or of a given color within a given region of the image. The major purpose of this Vision Tool is to determine the presence or absence of image features by looking at an area. The Vision Tool analyzes the pixels and determines how many of them are between an upper and lower threshold value for grayscale system or how many of them match the learned colors for a color system. Figure 29 shows an example of its use.

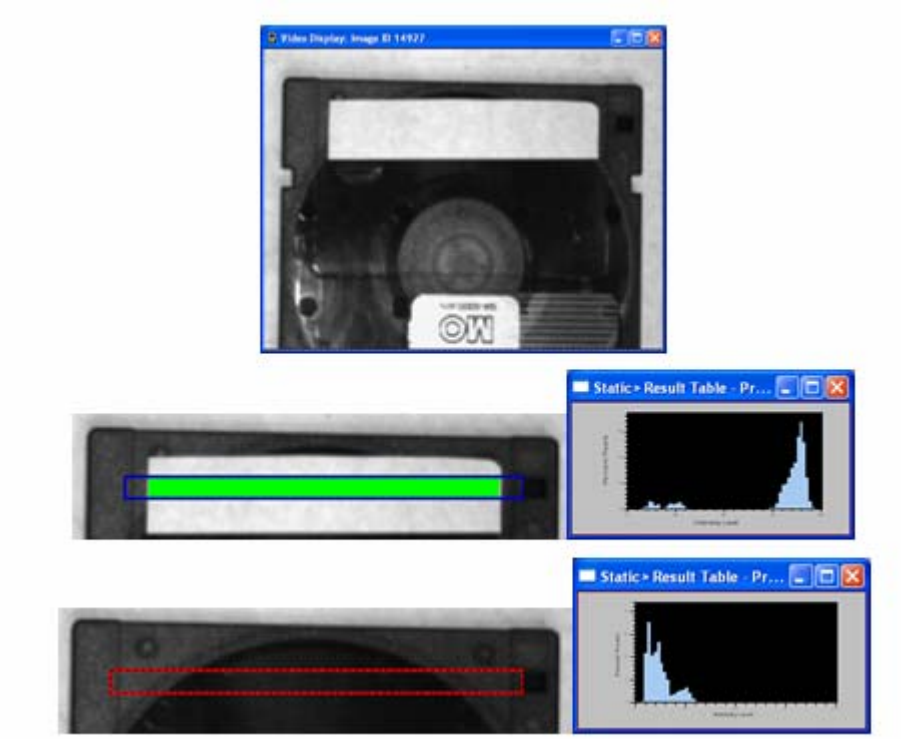


Figure 29: Use of a Pixel Counting Vision Tool to verify the presence and correct position of a label.

The Vision Tool in this case is detecting the label presence on the computer disk. This particular example uses a thin rectangular search area to limit the inspection area. The pixels within this area are grouped based on their intensity levels. The histograms, illustrate the pixel analysis of the image. For the first case, the label is present, so most pixels under the Vision Tool are light. This is reflected in the histogram which shows about ninety percent of the area above the minimum threshold and below the maximum threshold values. For the second case, all the pixels are dark (below the minimum threshold) so the Vision Tool reports 0%. This example only required the verification that the label reached a certain point, so the thin search area Vision Tool was adequate, but if the inspection consisted of the verification of the entire label, a larger rectangular area Vision Tool could be been drawn over the label.

This type of Vision Tool can be set to pass or fail based on the % of pixels matching the threshold criteria or learned colors. For the example above, a minimum % Pixel count should be established to determine what is acceptable and what is not. A minimum bright area of 90% would cause the Vision Tool to pass when the label is correctly placed and to fail when the label is incorrectly placed.

Blob Counting: Count in Area Vision Tools:

The Count in Area Vision Tool in Blob Counting mode is designed to find, count, or track blobs. Blobs in Intellect are defined as areas of connected pixels of similar intensity. Blob Counting is the fastest of the Count in Area Vision Tools but is the least tolerant of intensity changes and noise.

The Vision Tool in Blob Counting mode analyzes the image and isolates the blobs based on user options (threshold level and type of blob to locate). There are several intensity based threshold options. For an explanation of these options, refer to the section on thresholds. Once the blobs are located, the Vision Tool preprocessing options (Erode, Dilate, Close and Open) can be performed to eliminate some noise in the image. If the image requires more dedicated processing options, the user can reference dedicated PreProcessing Filter Vision Tools to help further condition the image before the Vision Tool inspects it. When the image contains noise, the number of blobs reported by the Vision Tool may be affected by it. When that happens, the Vision Tool becomes less reliable and the processing time for the Vision Tool

increases due to the extra blobs being processed. In order to overcome this type of situation, the user can limit the blob size. By doing this, blobs caused by noise (usually very small) can be filtered out of the Vision Tool. Using this size feature saves processing time since unwanted blobs are filtered out before any heavy computation is performed on the blobs.

This Vision Tool has a very specific image marking. White corresponds to the learned blob. Gray corresponds to blobs that are being counted. Boundary blobs (blobs that touch the edges of the Vision Tool) have no marking. In the case of the boundary blobs, the user selects whether those blobs should be counted or discarded. If the option is to count them, they will be marked in gray just like regular blobs. Blobs that have been discarded because of their size will have no marking.

The Selection parameters are used to isolate blobs created in the generation phase. The main use of this tab is to help the user find the blob(s) that the application requires by calculating a number of parameters about the blobs(s). If the objective is to track a part for rotation and translation, a particular blob must be isolated and its position must be calculated. When this is done, other Vision Tools can reference the blob selector for position reference. If the objective is simply to count the blobs or to verify their correct size/shape, the Vision Tool can do that automatically. The parameters that the Vision Tool can extract from every blob are: angle, position, area, bounding box size, eccentricity, compactness, perimeter, average intensity, and radius. For an explanation of these parameters see **Error! Reference source not found.** at the end of this section. The user can choose to use some or all of these parameters to help discriminate between the desired blobs and other blobs in the image. A match score is generated based on a combination of all the selected blob parameters. The Vision Tool can display the value of the selected parameters for every blob in the image.

Note: the user should use an Area Positioning Vision Tool if a part needs to be tracked and other Vision Tools will be repositioned based on the part location. However, if multiple position information is desired, the Count in Area Vision Tool in combination with a script or datalink transfer can be used to output multiple position information to other devices.

Figure 30 shows an example where a particular type of part is being counted based on its distinct blob characteristics. Four blobs are generated but only one meets the counting criteria.

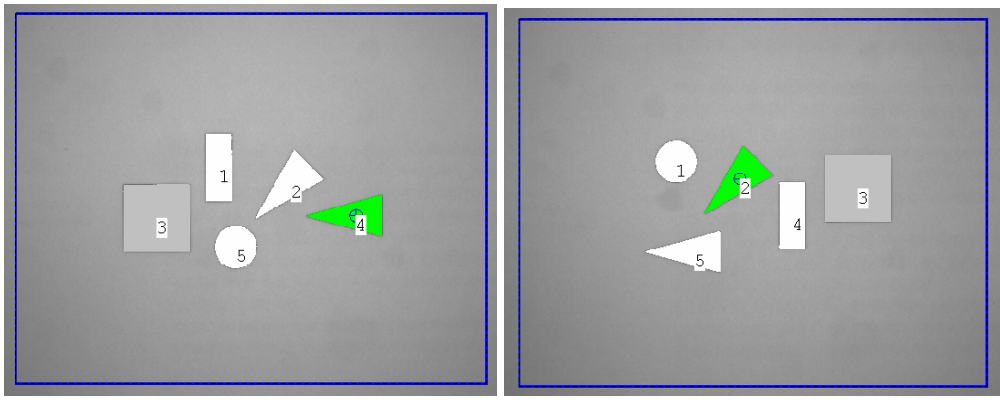


Figure 30: Use of the blob selector for position reference.

As the figure shows only one blob is being selected (highlighted in white). The image also shows the marking for the orientation of the part, the blue crosshair at the center of it. This blue crosshair has a tail to indicate the angular orientation of the part.

Figure shows an example where blob tools are used for flaw detection and part counting. The first image shows two pills. The second image shows the image marking for the blob tool when both pills are found. In this case the Vision Tool can be set to pass based on pill count.

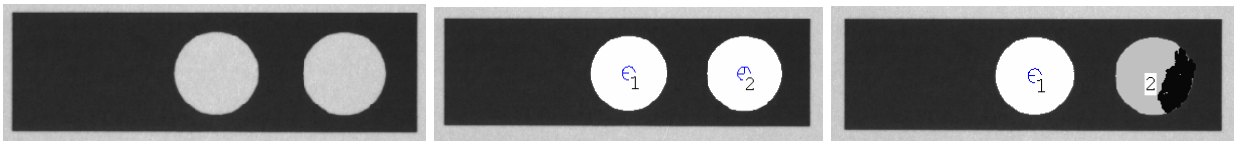
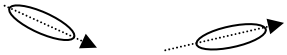
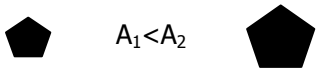
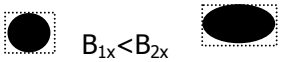
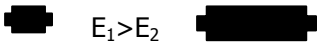
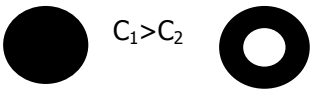
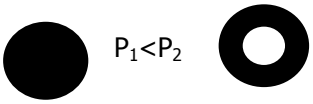
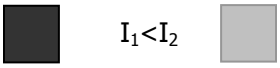
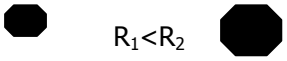


Figure 31: Example of use of blob tools for flaw detection / part counting.

The last image shows the marking for a case where one of the pills is defective. The damaged pill is discarded so the Vision Tool only finds 1 pill. In this case the Vision Tool should fail based on the blob count.

Table 7. Description of Blob Parameters

Parameter	Description	Example
Angle	Measured from clockwise from 3 o'clock position. Must check "Calculate Blob Angles" to then select blobs in range.	$A_1<A_2$ 
Position	Position of blob centroid as measured from image origin (upper left corner). Must check "Calculate Blob Positions" to then select blobs in a certain range.	Blob 1 (142, 83); Blob 2 (140, 96) $X_1>X_2$; $Y_1<Y_2$
Area	Measured in pixels. Selecting blobs based on area is a very common way to sort shapes.	 $A_1<A_2$
Bounding Box	BB is the width and height of the rectangle that encloses a blob.	 $B_{1x}<B_{2x}$
Eccentricity	Measures rotational symmetry of a blob. 0-1 scale with most eccentric shapes near 0 (very asymmetrical) and least eccentric near 1 (very symmetrical). Independent of scale.	 $E_1>E_2$
Compactness	A measure of blob density. 0-1 scale with least compact shapes near 0 (less dense) and most compact near 1 (more dense). Independent of scale.	 $C_1>C_2$
Perimeter	Measures the total length of all edges around the blob, including interior edges.	 $P_1<P_2$
Avg. Intensity	Measures average intensity value of the connected pixels.	 $I_1<I_2$

Radius	The radius in pixels of a shape. For non-circular blobs, this value represents the distance from the center of the blob to the point that is farthest away from it.	 $R_1<R_2$

Object Counting: Count in Area Vision Tools:

The Count in Area Vision Tool in Object Locate mode use advanced algorithms to learn the edge characteristics of an object and find occurrences of that object in subsequent images. The Object Locating mode can be applied in a variety of Vision Sensor applications, including:

- **Flexible feeding, motion control, robotics:** Parts could be presented in any orientation and may overlap.
- **Short product runs:** The key to these applications is quick setup because of the variety of parts manufactured.
- **Position finding/referencing:** The Object Locating modes learning ability means quicker.

When the Object Locate mode is selected the Vision Tool extracts edge segments of the part that are contained in the learn region of the Vision Tool. The edge segments describe the geometry of the object. Since the Tool uses edges it is less sensitive to lighting and intensity changes in the image making it very robust. However, the increased robustness of the Vision Tool comes at the expense of inspection time. The list of features is kept in memory and the Vision Tool uses that list as the learned model. In subsequent images, the algorithm searches for the features contained in the model to determine the presence of the learned object. The double area Vision Tool uses two areas. The inner area is for learning the object and should be drawn just outside the object. The outer area indicates the region to search for the object.

Object Locate Vision Tools have a number of parameters that the user can set to better find the objects. In order to minimize processing time, the user can limit the movement that the objects are to have. That is, when the application allows for it, the user could limit the movement in the X and Y directions as well as rotation and scale, so the search for the object does not take much processing time. It is for these cases that the Vision Tool offers a scaling tolerance parameter which the user can select to identify objects when they are proportionally bigger or smaller than the learned model.

Another important parameter that Object Locate Vision Tools have is flexibility. The user can set the Vision Tool to be more or less flexible. The Vision Tool should be less flexible when the object to be located does not vary in size or geometry. If the parts are neither defective nor overlapping, the Vision Tool should only recognize the presence of an object when many features of the learned model are present in the area. This option makes the Vision Tool less susceptible to noise. For the opposite case, when objects tend to overlap or even change in geometry, the Vision Tool should be made more flexible. This will allow the Vision Tool to find objects where there are only a few characteristics of the learned model. The user should keep in mind that the more flexible the Vision Tool is, the more susceptible to noise it becomes. That is, if background noise or even other parts in the image present some of the features that the learned model contains, those regions will be identified as a new object. The parameters that allow the Vision Tool to be more or less flexible are:

- **Scale Tolerance:** Allows the Vision Tool to find bigger and smaller objects
- **Minimum Match Score**: These indicate a minimum percent of features and perimeter that must match the learned model for the object to be indicated as a match
- **Locate Pixel Distance:** The pixel distance refers to the acceptable distance of the found edge from the model edge. Increasing this distance will make the Vision Tool less sensitive and increase the match score.

- **Fine –** Coarse Slider: You can select the resolution at which Object Locate finds the part at. The default setting is "Fine". Medium resolution is ¼ of the original area, and "Coarse" is 1/8. As you go lower in resolution, the effect of noise becomes smaller, but the accuracy in the found location becomes lower. With "Auto" checked, it starts at the coarse resolution and then fine tunes the position of the found object all the way up to the fine resolution. If you have the slider at Medium resolution with "Auto" checked, it stops at medium resolution. With "Auto" resolution in 1.3, OL becomes more reliable and faster at the same time.

All these categories consist of a number of different parameters, for explanations about each one and how to change them please refer to the Intellect help files. To understand how much to change the values, please refer to the Vision Tool properties: under "Model Object", a number of parameters are displayed corresponding to every object that is found in the image.

Image marking for this Vision Tool is similar from the others. The user can customize it by choosing among two options: Object Marking and Live Edge Image. In Object Marking, the found object is marked with edges that matched the learned model in green and missing edges of the learned object in red. When displaying the Live Edge Image, the relative strength of the extracted edges is show. The brighter the intensity the stronger the edge.

This Vision Tool has many applications and uses including position feedback for another Vision Tool or to a robot or other external device. However, if the application consists of counting the parts, the Vision Tool can be set to pass or fail based on the number of objects found.

Pattern Counting: Count in Area Vision Tools:

The Count in Area Vision Tool in Pattern Match mode is used to check pixel patterns against a learned intensity template. This mode uses a normalized grayscale correlation algorithm to find the best match of a learned pattern in the search region. Therefore, it is valuable when strong edges are not present, when intensity changes occur and when large amounts of noise may be present. However, it is a computationally intensive Vision Tool which should only be used when other Vision Tools do not provide an adequate inspection.

NOTE: Although this Vision Tool can be attached to a positioning Vision Tool to inspect the correct location, it can only translate. That is, the Vision Tool will not inspect the part correctly if it is rotated.

Positioning Vision Tools

This group of Vision Tools helps to locate parts and features when working with applications that need to account for part movement. These Vision Tools are:

- Position along a Line
    - Line Mode (Translation): These Vision Tools look for edges or patterns along a certain path or in an area and then provide offsets in the horizontal and/or vertical direction that would serve as position reference for other Vision Tools.
    - Arc Mode (Rotation): these Vision Tools look for an edge or pattern along circular or elliptical paths. They would provide angle information that can be used as position reference for other Vision Tools.
- Area Positioning
    - Blob Locating: Finds the location of a part or area of a part based on blob parameters.

- Object Locating: Finds the location of a part or area of a part based on the edges o f the parts use geometric edge location.
- Pattern Locating: Finds the location of a part or area of a part base on the intensity pattern representing the part using normalized grayscale correlation.

- Line Fit: Specialty positioning Vision Tool used to find the location of a straight edge by finding edges and fitting a line to the found edges. This Vision Tool reports a rotation angle.
- Circle Fit: Specialty positioning Vision Tool used to find the location of the center of a circle.

It is important to note that these are not the only Vision Tools that could be used for providing a position reference for other tools. They are just the simplest and easiest positioning tools available. For more information on which Vision Tool to choose or how to approach applications where part tracking is needed, consult the section Dealing with Part Movement in Chapter 4

Position along a line / arc

Position along a line or arc Vision Tools locate the absolute position of an element in an image in cases where that element's position changes between inspections. There are two types of shapes: Lines and Arc. Lines find detect translational movement and arcs detect rotational movements.

Translational lines determine the position of an edge or a pattern (edges and patterns were explained under Count along a Line Vision Tools). The option to locate patterns should be used when the application allows for it because it makes the Vision Tool less sensitive to noise. This type of Vision Tool reports the distance from the origin of the Vision Tool. It searches the pixels from the origin in the direction of scanning (set by the direction in which the Vision Tool was drawn). When it finds the edge/pattern that the user indicated, it reports the distance from the origin of the Vision Tool to the location of the edge/feature in pixels. Figure 32 shows a line Vision Tool and its pixel graph.

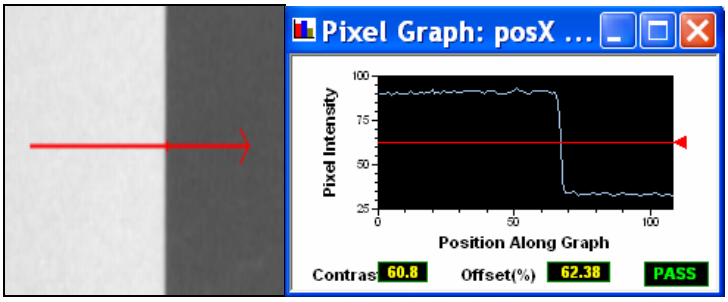


Figure 32: Position along a line Vision Tool and its pixel graph.

This Vision Tool can only corrects for movement in one direction. However, in most cases the part needs to be tracked in two directions of translation. When that is the case, another Vision Tool of this type needs to be created and a reference between them needs to be assigned. That is, the user needs to create two Translation Vision Tools: one to determine the translation in one direction and a second one to determine the translation in another direction. The second postion tool should reference the first position tool and all other inspection Vision Tools should reference the second position tool. The creates the necessary position reference chain mentions earlier.

When looking for an outer most point of a curved edge, increase the Scan Line Width in the Shape tab to get the effect shown in Figure 32.

Besides providing a position reference for other Vision Tools to perform the inspection, Translation Vision Tools can be used to inspect a part. In applications where the position of a certain feature decides whether the part passes or fails, translation Vision Tools can be used to actually determine the final result. They can be set to pass or fail based on how much the part has moved in the image.

Rotation with Vision Tools: Arc and Line Fit

When the Position along Line Vision Tool is drawn as an arc, the Vision Tool computes the angle of rotation of objects in the image. Like line Vision Tools, arc Vision Tools allow different positions of an object without failing the part for being in an unexpected position or location. In other words, other Vision Tools can reference the output of Rotational Vision Tools in the same manner as with line Vision Tools. There are two types of Rotation Vision Tools: arc-based and parallelogram-based.

Arc-Based Vision Tools are often applied when the position of an object is known to have fixed radius and center of rotation. For example, the ball bearings in a race can be moved around a circle, but are restricted from moving up and down independently. A Circular Arc can be drawn to locate one bearing and other Vision Tools can reference the position output of the Circular Arc. Basically, the arcs used to track rotations work like translation lines do. They search along the arc for a specific edge or a specific feature. Then, they report the percent offset from the origin of the Vision Tool. The difference with translation lines is that besides that percent offset, there is a center of rotation associated with the angle being reported. That center coincides with the center of the tool as drawn. Thus, when this tool is used to track the rotation of a part, the closer the center of the tool is to the true center of rotation of the part, the more accurate the results. Figure shows an example where the incorrect placement of the rotation tool causes a problem with the inspection. In the upper left image, we see an intensity tool at the tip of an arrow. Since the center of the arc does not coincide with the center of rotation of the arrow, as the arrow changes position, the arc tracks it incorrectly. It finds it but it uses its own center as the center of rotation. In the third image the arc is correctly placed, so as the arrow changes position, the arc tracks it correctly and it passes a good reference for the intensity Vision Tool which performs the inspection at the tip of the arrow as expected.

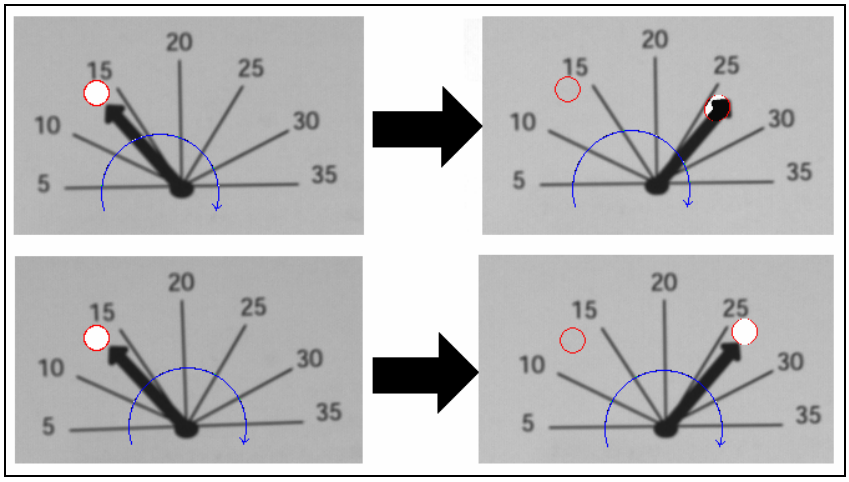


Figure 33: Incorrect and correct uses for a rotational arc.

The second Vision Tool which reports rotation is the Line Fit Vision Tool (this is a rectangle-based Vision Tool that can be rotated). This Vision Tool is used when an object with straight edges is not fixed in orientation. The Vision Tool fits a line along the straight edge and tracks the rotation of the line. A comparison between the arc and the parallelogram Line Fit is shown in Figure .

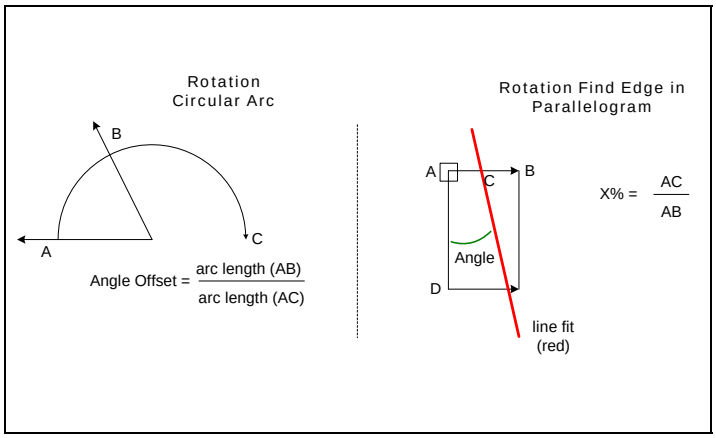


Figure 34: Output calculation for Arc (left) and Parallelogram (right) Rotation Vision Tools.

As the figure indicates, besides the angle of rotation the parallelogram outputs a distance. That distance, illustrated as X% in the figure, is obtained from the origin line of the Vision Tool. That is, the first line that the user draws for this Vision Tool acts as a Translation Vision Tool. Furthermore, the direction in which that line is drawn indicates the direction of the scanning lines. This Vision Tool consists of a number of parallel scanning lines. Those lines look for edges or features (according to user specifications) and each one determines a point where they intersect the selected edge/feature. The Vision Tool then fits a line along those points and tracks its rotation. The intersection of that line with the original Vision Tool line is used for translation output. Figure 35 shows a sample image where the parallelogram tool is used. The square in the upper left corner of the tool indicates where the user started drawing the Vision Tool. The arrows indicate the direction of scanning, so this tool was drawn starting on the upper left corner going down, then it was expanded to the left to indicate the width.

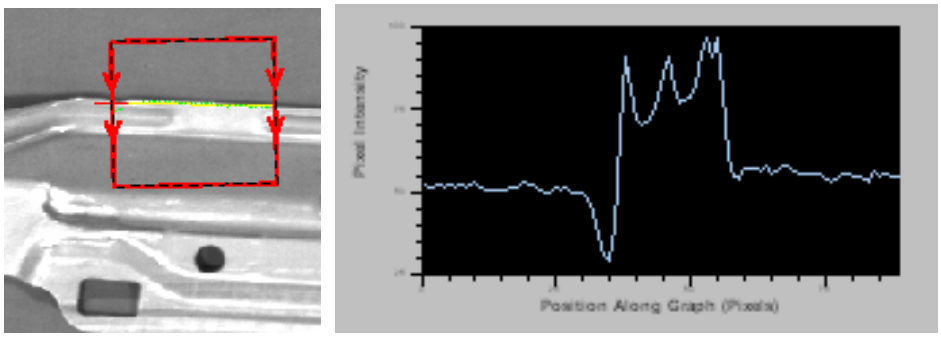


Figure 35: Use of the Parallelogram tool of the Rotation Vision Tools. The pixel graph associated with this Vision Tool illustrates the pixels along the origin line (left edge in this case).

The left line then becomes the origin line and works like a Translation Vision Tool. The Vision Tool pixel graph corresponds to the origin line. As it was discussed before, pixel graphs are only compatible with line Vision Tool; however, in this case we have an area Vision Tool and a pixel graph associated with it. Since the most valuable region of the Vision Tool is the origin line, that is the line used for the pixel graph. The pixel graph then shows a numerical value for both the angle of the part and the position of the edge or feature.

Area Positioning Vision Tools

The Area Position Vision Tools search in an area for the location of a single object or feature that is trained. This Vision Tool uses one of three different algorithms to accomplish the locating of an

object/feature; Object Locating, Pattern Matching, or Blob Analysis. The position of the found object can be used for moving other Vision Tools when the inspected part moves in the field of view or it can be used in combination with a coordinate transformation to give the coordinates of a part to a robotic device.

When first drawn, the Area Position Vision Tool will default to the Blob algorithm and the Vision Tool will automatically learn the object under the rectangle that is first drawn. A larger search region rectangle will automatically be drawn larger than the learn area rectangle. To change the Algorithm used by the Vision Tool, change the "Select task:" on the Options page of the Vision Tool Parameters.

Blob Locate

The Blob Area Position Vision Tool uses standard and advanced thresholding and automatic parameter setting to find the location of an object based on connected areas of similar intensity pixels. It is the quickest of the area based position Vision Tools. A Selection editor makes setting blob parameters for the desired object easy. This Vision Tool uses the same principles as the Count in Area Vision Tool in Blob Counting mode except, the Positioning Vision Tool limits the number of found Blob to one.

Object Locate

The Object Locate Vision Tool uses advanced edge extraction and analysis to locate an object any where within a search area. It can locate both whole objects as well as features associated with the object. It can also find partial occluded and touching objects. It is insensitive to non-uniform lighting and low illumination. An Object Model editor makes manipulation of the trained object to locate easy. This Vision Tool uses the same principles as the Count in Area Vision Tool in Object Locate mode except, the Positioning Vision Tool limits the number of found Objects to one. There is also a "First Match – Best Match" slider in this mode which is set to "First Match" by default. The more you move the slider to the right, the more model segments and algorithms it will use for matching. If you have a case where the location of the object is not found correctly, you can move the slider to the right or increase "Minimum Match Score".

Pattern Match Locate

The Pattern Match Vision Tool uses advanced gray scale correlation to locate a pattern anywhere within a search area. It locates the position of a learned pattern that translates. It finds the pattern with the highest match score based on a gray scale correlation. Although it can tolerate some limited rotation it is designed more for finding object in pure translation without rotation. If location of a part that rotates is desired, then the Object Locate or Blob Positioning Vision Tools are recommended. This Vision Tool uses the same principles as the Count in Area Vision Tool in Pattern Match mode except, the Positioning Vision Tool limits the number of found Patterns to one.

Vision Tools used for Specific Inspections

The next group of Vision Tools is the Specific Inspection Vision Tools set. These Vision Tools perform many tasks from very specific inspections to a user defined task (programmable). These Vision Tools are:

- Filter Vision Tools: The Filter Vision Tools use powerful preprocessing algorithms to manipulate the pixels of a region or the entire field of view to help enhance the inspection of other Vision Tools.
- Measurement Vision Tools: Take measurements from parts for inspection or simply to gather data.
- Math Tools: Perform mathematical calculations based on information from the measurement Vision Tools.

- Identification: Identifies the part based on multiple learned models using either blob properties or color properties.
- Readers: Perform readings of 1D and 2D codes as well as OCR (Optical Character Recognition). They can be used for recognition, verification, and in some cases for grading.
- Color Pixel Counting: Color matching tools. Learn colors and determine their presence in subsequent images.
- Defect Detection: Learn a model and then verify that the model is present in subsequent images, if the image does not match the model it reports the amount of error.
- Color Monitoring: Ultra precise color verification tools.
- Script: Programmable tool that can access other Vision Tools for data gathering, etc.
- Spectrograph: Available in the DVT Spectral Sensor, this Vision Tool analyzes the full spectrum of a part or light source and can compare it against a list of previously learned spectrums.

Preprocessing Vision Tool

The Preprocessing Vision Tool category contains Vision Tools that alter the image in some way to help the other inspection Vision Tools accomplish their tasks most effectively. In general, the preprocessing Vision Tools reduce background noise, increase edge or foreground contrast, or move image pixels around. Below is a list of filters in the Preprocessing Vision Tool.

The Filter Vision Tools use powerful preprocessing algorithms to manipulate the pixels of a region or the entire field of view to help enhance the inspection of other Vision Tools. They can be a very powerful tool to help reduce background noise, enhance edges, filter colors and other operations. The operations can result in a more robust inspection at the expense of inspection time. Since the filtering algorithms are often complex and are applied to an area, they will increase the overall inspection time.

To use a filter Vision Tool, draw the preprocessing Vision Tool in the area in which your inspection Vision Tools will be inspecting and select the desired preprocessing operation in the Options dialog. In order for the other inspection Vision Tools to use this filter, the inspection Vision Tool must reference the filter Vision Tool in the General Section of its Vision Tool parameter page in the image reference dropdown. The filter Vision Tool rectangle must encompass the entire inspection Vision Tools shape. If it does not or if the inspection Vision Tool move off of the filter Vision Tool, then the inspection Vision Tool will truncate the shape in most cases to fit the filtered image or fail with a "moved off image" failure cause.

When first drawn, the Filter Vision Tool will default to No Filter as the Preprocessor Operation. To change the Algorithm used by the Vision Tool, change the "Preprocessor Operation:" on the Options page of the Vision Tool Parameters. If more than one Preprocessing operation is desired, the Aggregate operation can be used to concatenate the operations with one Vision Tool.

- Morphology – Performs dilation, erosion, open or close operations to help eliminate small holes in an image to remove small noise pixels.
- Noise Reduction – Smoothens the image by performing low pass or median operations. Effective at reducing small noise pixels however it also causes a blurring effect.
- Gradient – High pass filter type operations highlight edges that are sharp in the image. This can be useful when the edges are sharp but the lighting is non-uniform.
- Thresholding – The Thresholding filter is used to suppress areas of the image based on the filter criteria by turning them to a black intensity so that they can be ignored. The Color filter allows a specific color or colors to be eliminated and set to black or for the background to be suppressed to black. The Edge filter will eliminate edges by setting the edge pixels to a background black color. The Blob Filter allows connected regions of pixels to be ignored or passed through. Segmentation sets edge pixels to black background and finds segments enclosed by edge pixels.

- Enhancement – The Enhance filter group accentuates the image by providing different effects like extending the dynamic range of the intensities, darkening the edges, suppressing or enhancing color planes, etc.
- FOV Balance – Used to compensate for consistent non-uniform lighting. Used most commonly for Reader applications.
- Subtract Dark Image – Subtracts the live image from the stored dark image. It is usually used to remove dark current CCD noise. Can also be used as image compare.
- Flip Image – Always the image to be flipped along an image axis. Mainly used when the image is formed with a mirror.

Measurement Vision Tools

Measurement Vision Tools are used to take measurements from the parts being inspected. These measurements will be in pixels unless a scale factor or coordinate transformation is performed. There are two types of Measurement Vision Tools: line based and area based. In this case, the area based Vision Tools consist of a number of scanning lines. That is, they collect data from many lines in order to produce a more accurate result. Typically, each scan line will produce a data point by finding an edge. The Vision Tool then uses all those data points to perform calculations (e.g. fit a line or a circle). All the Measurement Vision Tools can use gradient based methods for thresholds in addition to the intensity based types. When a gradient based threshold is being used, sub-pixel accuracy can be easily achieved and some advanced Vision Tool parameters become available: Edge Width, Use Best Edge, and Hough Transformation. These options make the Vision Tool less sensitive to noise. This is how they help the Vision Tool perform the inspection:

- Edge Width: This option lets the user indicate how many pixels determine the edges to be used. If the images are not well focused, the edges will not appear as sharp. In those cases, the user needs to give the Vision Tool an idea of the transition width to better calculate edge positions. Figure 36 illustrates the use of this option. The Vision Tool on the top edge does not find the fuzzy edge with the default edge width setting. The bottom vision tool's edge width setting is set much higher.

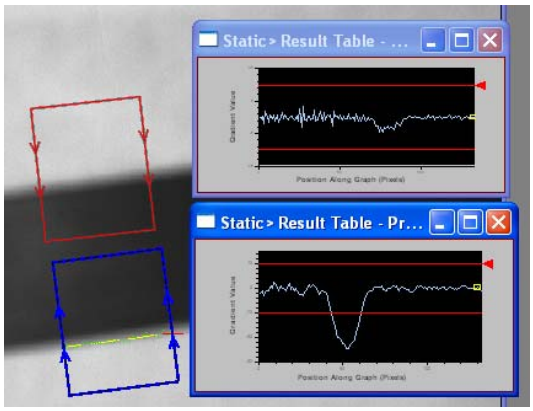


Figure 36: Area Measurement Vision Tools in presence of fuzzy edges. The gradient graphs indicate the width of the transition.

- Use Best Edge: This option is for noisy images. Sometimes background noise produces minor peaks in the gradient graph. In cases where those peaks happen to reach levels above threshold, this option will discard them and keep only the highest peak. The use of the Best Edge option is illustrated in Figure 37. In this example, if the option was not selected, the Vision Tool would report the edge close to pixel number 20, which is where the first peak (above threshold) is located. The use of this option makes it report the true edge instead because it is a better defined edge. The Vision Tool on the left is not using this advanced option, and the data points

generated by the presence of noise cause the output line to be calculated at an incorrect location. The second Vision Tool uses this option and the output line is correctly drawn over the edge of the part.

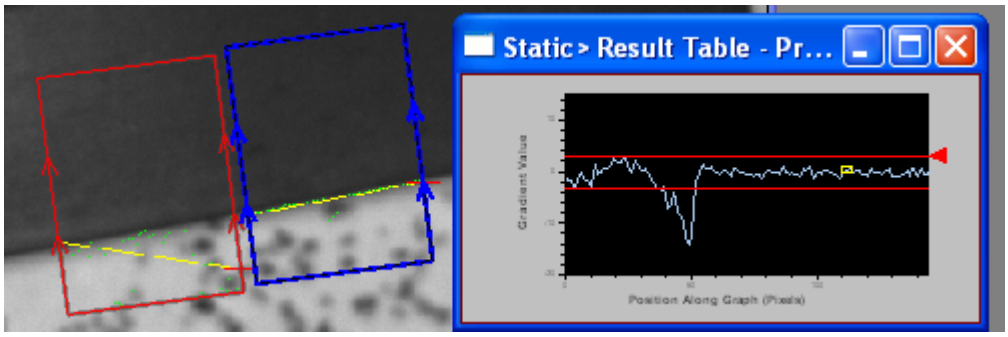


Figure 37: Effect of the use of the best edge option. The Vision Tool on the left fails to locate the edge because of the background noise, whereas the one on the right looks for the best edge (the highest peak in the gradient graph).

All these options increase the processing time of the inspection, so if timing is an important issue, the user should first try to avoid them by changing the light or the lens being used to improve the image.

If sub-pixel accuracy is not a requirement, the Vision Tools can use intensity based thresholds. This option may increase repeatability in the measurement and it adds some advanced options to deal with noisy images as well. For more information on these operations see Processing Operations under Vision Tool Parameters.

Besides the specific parameters discussed so far, some measurement tools include extra options to make the Vision Tool more precise. When area based Measurement Vision Tools are being used, more parameters become available: Scan Density, Max Iterations, and Outlier Distance.

- The Max Iterations option lets the user indicate how many times to recalculate the line/circle being interpolated using the data points. After every calculation, points are assigned a certain weight for the next calculation based on the distance from them to the computed output (line/circle). By doing this, the algorithm puts more emphasis of true edge points and less on noise. Figure 38 illustrates the use of this option. Image (a) shows the output line calculated with no iterations. As the image illustrates the output line is not on the edge because of the imperfection of the part. Images (b) and (c) show the same Vision Tool after 2 and 5 iterations. Image (c) is the only one that finds the edge of the part regardless of the imperfections that the part presents.

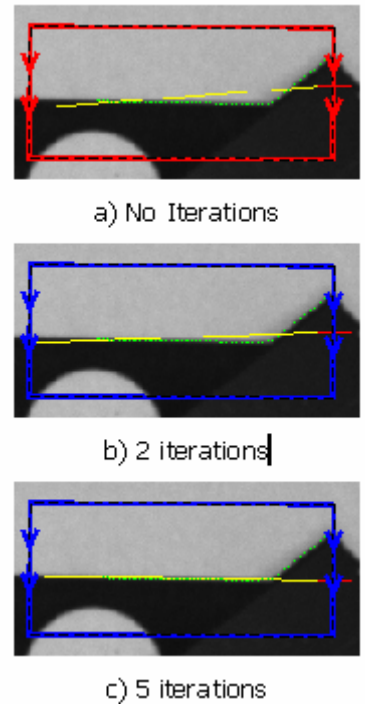


Figure 38: Use of the number of iterations to overcome imperfections/noise in the part. Note: scan lines show for illustration purposes only.

- The Outlier Distance option is used after the iterations and it discards points that are more than a user defined number of pixels away from the output line/circle. This option helps find the right edge of the part giving a more accurate result.

There are five main measurements that the Vision Tools can compute: thicknesses, radius, straightness, roundness and concentricity.

Thickness measurements can be performed using a line or an area Vision Tool. In both cases the Vision Tool can scan inside out or outside in (just like using a caliper). The scanning lines start scanning from the ends of the Vision Tool or from the center until they find the specified characteristic. This characteristic could be a peak value in the gradient graph (when using gradient computation) or a specific type of edge (when the threshold type is intensity based). The output of the Vision Tool is the distance in pixels between the edges of the part. Figure 39 shows the use of both line and area based Vision Tools. Line Vision Tools (called Measure Across Line) can only measure the distance along the line.

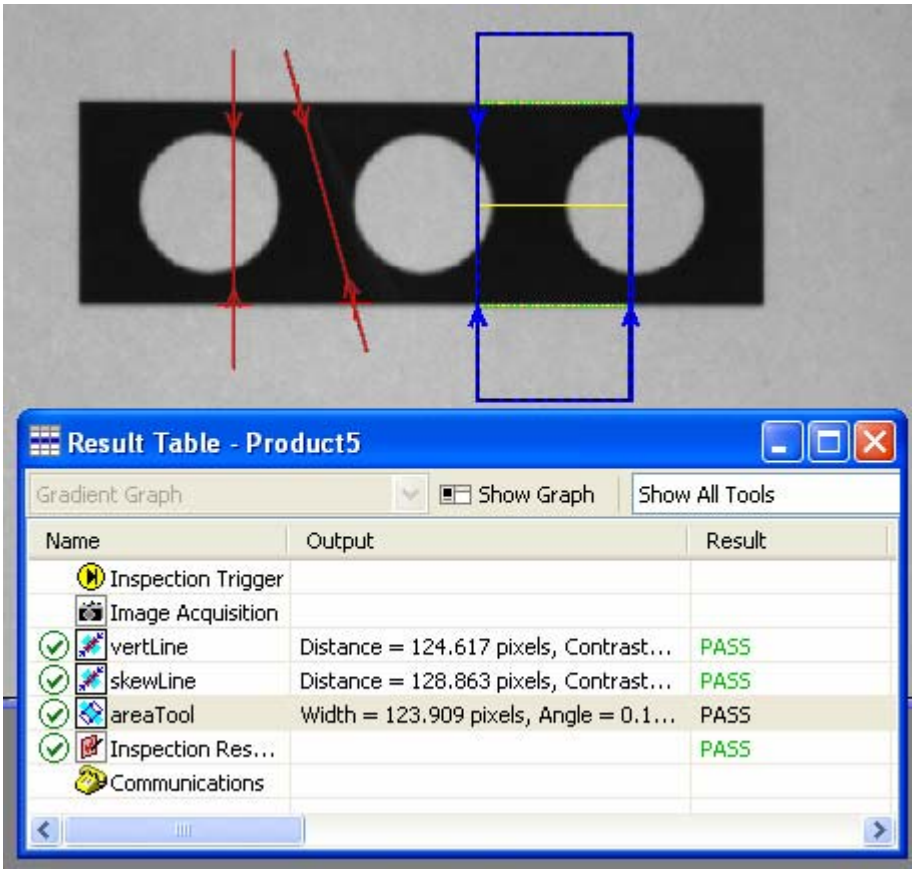


Figure 39: Caliper type Vision Tools measuring the height of a part.

As the figure illustrates, more precision can be achieved using the area Vision Tool (called Measure in Area). Line Vision Tools require that the Vision Tool be drawn at a perfect 90-degree angle with respect to the edge of the part. None of the line Vision Tools in the image is drawn in such way, so the reported distance is greater than the actual distance. Area Vision Tools on the other hand use many scanning lines and each scanning line provides a data point. The algorithm then fits a line along each set of points and reports the distance between those lines as well as the angle between them. If the angle is other than zero degrees, the algorithm averages the maximum and minimum distances that it calculates from the data points in the Vision Tool. The area Vision Tool requires more processing time because it not only consists of many scanning lines, but it also requires further processing for the line fitting calculation. One way to reduce the processing time is to reduce the scan density in the Shape Parameters page. Note that if the Scan Direction is set to Inside Out and the center line of the Vision Tool is not inside the part, it will fail.

The Measurement Vision Tools that calculate radii are the Measure in Circle. They have the same functionality; they differ only on their applications. A Find Circle tool should be used when the circle to be measured is a complete circle. The segmented version of this tool is used when there is not a full circle. Figure 40 illustrates the use of these two different tools.

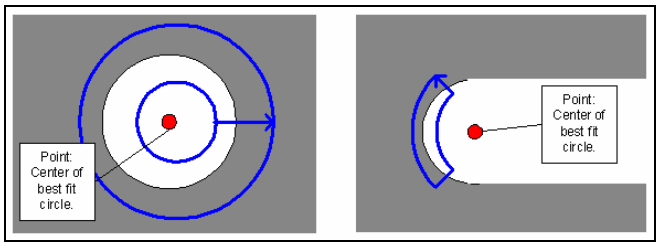


Figure 40: Different uses of Find Circle and Segmented Find Circle tools.

To use these tools the user draws two circles, one inside the part and one outside. Then, scanning lines will go from the first circle drawn towards the second one looking for the edges of the part. The intersections of the scanning lines with the edge of the part determine data points. The algorithm then fits a circle through those points.

Different outputs can be obtained from the area based tools. The user can select minimum, maximum, median or average distance (or radius, depending on the tool being used). These tools can be set to pass or fail based on specifications for the distances being measured (min radius, max offset, etc.) when the inspection consists of the determination of a correct size. Another feature that the user can select for quality control is the overall roundness/straightness. Depending on the area tool being used, there will be an extra output that indicates how spread out around the output line/circle the data points are. In other words, straightness (or roundness) is given by the sum of the distances between the data point that is farthest away in one direction and the one farthest away in the other direction of the result line (or circle). Hence, a perfect edge would produce a very small straightness (roundness for a circle) whereas a noisy edge would produce a high straightness (roundness for a circle). In this case, the outlier distance helps a lot in presence of noise. When noise produces data points for the final line, the straightness can increase considerably. That is, one single data point caused by noise far from the edge will increase the straightness dramatically. Outlier distance will discard those so the calculation for straightness concentrates on true edge data points instead of noise. Figure 41 illustrates the use of this option. The figure shows a dark part and a noisy background. The dark pixels of the background cause some scan lines to identify data points far from the edge.

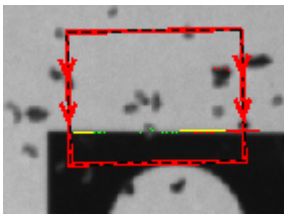


Figure 41: Effect of Outlier distance on Straightness for noisy images

When a number of iterations are selected to correct that effect, the edge is correctly located. However, the data points caused by the noise are still active, so the Vision Tool reports a straightness of 49 pixels. When the outlier distance is used to discard points located more than 5 pixels away from the line the data points produced by the noise are discarded and the straightness drops down to 1.92 pixels. This type of pass/fail condition could be used to verify the quality of a cut or a drilling in a part.

Another use for the Measurement Vision Tools is positioning. Even though Intellect has a dedicated number of Vision Tools for position reference, Measurement Vision Tools can provide a position reference as well. In cases where the positioning of a part needs to be very precise, the user can select a Measurement Vision Tool instead of a Positioning Vision Tool as the position reference.

Math Operations: Points and Lines, Statistics Vision Tools

Common math operations are performed on Vision Tool results by the Measure with Points and Lines Vision Tool and Statistics Vision Tool. The Measure with Points and Lines Vision Tools offers an assortment of Vision Tools. Unlike any other tools, except Scripts, these Vision Tools require no additional drawings. These Vision Tools reference other Vision Tools that have been previously defined in the Inspection. Usually, they reference Measurement or other Math Vision Tools using their outputs (lines and points) for the calculations. The Points and Lines Vision Tool calculates a distance or angle based on lines and points generated from other Vision Tools or it constructs new points or lines based on the references. Table shows a list of these operations and indicates the type of reference that the Vision Tool needs and the output it produces.

Table 8: Summary of math operations in the Points and Line Vision Tool.

Math Tool	References	Returns
Distance	Two points	The distance between the reference points
Angle	Two lines	The distance between the reference lines
	A point and a line	The perpendicular distance from the reference point to the reference line
	Two lines	The angle between the reference lines
Intersection	Two lines	The coordinates of the intersection of the reference lines
Midpoint	Two points	The coordinates of the midpoint between the reference points
Midline	A point and a line	The coordinates of the midpoint between the reference point and the reference line
	A point and a line	A line parallel to the reference line through the midpoint of the reference point and the reference line
Line through two points	Two lines	The bisector of the angle determined by the reference lines
	Two points	A line through the reference points
Perpendicular Line	A point and a line	The perpendicular to the reference line through the reference point

Defect Detection Vision Tool

The Defect Detection is a Flaw Detection Vision Tool that can be used to easily identify detects in a part or object that is consistently placed part or a position referenced from another Vision Tool. This Vision Tool learns the pixel pattern contained in the rectangle, polygon, or ellipse and compares the subsequent images to the learned one. Figure 42 shows an image sequence where the part that is present in the first image is being inspected for errors.

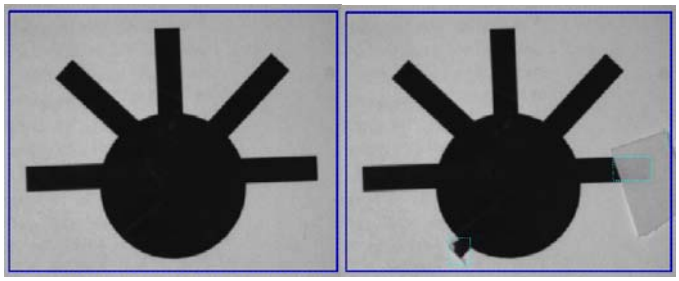


Figure 42: Sample images of defect detection vision tool with extra material and missing material.

In the example, the Defect Detection Vision Tool learns the first image and detects defects characterized by either extra information in the background or missing portions of the part. As the Vision Tool inspects

the second image, it finds the extra feature in the lower left section of the part. It also detects the missing portion on the right side of the part.

A powerful feature of this Vision Tool is its ability to ignore areas around the edges by using an image mask. The image mask places a lower weight on areas around the edges since they are likely to change from image to image. The mask can be created by multiple images and it can be expanded or shrunk manually. Therefore, the Vision Tool is less likely to report errors near the edges.

The Defect Detection Vision Tool can be set to pass or fail based on the total amount of errors detected inside the Vision Tool or by the number of defects of a certain size. It should be noted that the resulting error image of this Vision Tool can be used as an image reference by other Vision Tools (like blob counting) so that the flaws can be further processed. But user needs to make sure there is no fail condition set for the defect detection tool or image referencing would fail.

Identification Vision Tools

The Identification Vision Tools are used to recognize or identify a particular part or feature from a set of learned parts or features. There are two types of Identification Vision Tools: Identification and Color Identification.

The Identification Vision Tool uses blob based properties associated with an object to identify it from other learned objects. The Vision Tool finds the object that most closely matches the learned object based on different identification criteria like area, eccentricity, compactness, perimeter, intensity, radius, and color.

The Color Identification Vision Tool is used to identify an object based on the color content under the Vision Tool. The colors found by the Vision Tool are compared against the colors contained in the trained color models.

Readers

Intellect contains a number of Vision Tools dedicated to reading codes, labels, etc. These are the Reader Vision Tools. These Vision Tools consist of the 1D Barcode Reader, 2D Reader (DataMatrix, SnowFlake and Vericode) and OCR Vision Tool (used for Optical Character Recognition and Optical Character Verification).

One Dimensional Barcode Reader

The 1D Barcode reader consists of two types of Vision Tools: a line Vision Tool (arc or straight) and an area based Vision Tool. The types of barcode that can be read are Interleaved 2 of 5, USS-128, USS-39 (Code 39), UPC/EAN (found on most consumer products), Codabar, Code 93, POSTNET, PLANET, PharmaCode, BC412, PDF417, Micro PDF417, RSS-14 (regular, truncated, stacked and stacked omnidirectional), RSS Limited, RSS Expanded (regular and stacked), RSS-14 Composite, RSS Limited Composite, RSS Expanded Composite, UPC/EAN Composite and UCC/EAN-128 Composite. Note, some o f these codes can only be read with the area based reader.

The line Vision Tool (arc or straight) must be drawn through the barcode. Both versions of the Vision Tool have an extra option to include sub-pixel calculation, making it more powerful when the image quality is not the best and the line reader crosses the barcode at an angle other than 0º or 45º. This option makes the Vision Tool more reliable, and the impact on processing times is negligible so it should be used almost every time the above conditions exist. The Vision Tool can be programmed to auto detect the type of barcode (if the code is one of the first eight codes mentioned above) being read or to always search for a specific type of barcode. The automatic option adds some processing time and should be used only if the barcode type is unknown. The second option can be set manually (the user indicates which type of code is to be read) or automatically (sets the type of barcode upon read of the first instance of the barcode). Usually, applications need to read a certain type code which does not change,

what changes the encoded message that the Vision Tool needs to be read. Therefore, the usual setup for this Vision Tool is to set the barcode type upon read (auto-detect) if the user expects one of the seven types mentioned above; otherwise the type should be specified. Figure 43 shows sample barcodes. PDF417 is the one on the left; it can be observed that it is a very different type of barcode: it is a stacked barcode. When this is the case, the Vision Tool creates more scanning lines to read the different levels of the stacked barcode.

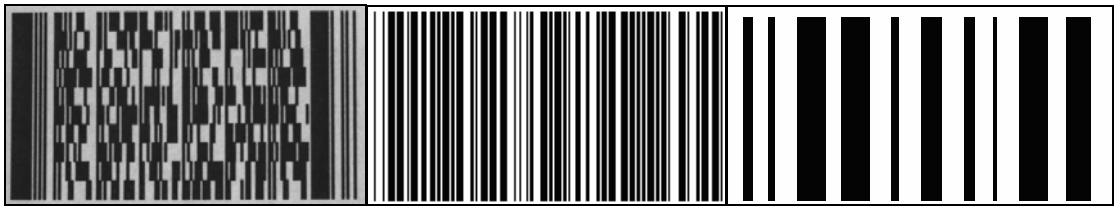


Figure 43: Comparison of PDF417, Code 39, and PharmaCode types. The first one is a stacked barcode type.

The last two barcodes in the image are Code 39 and PharmaCode. For most users it is impossible to visually determine the type of code being used because they have very similar appearance. In such cases the user should rely on the Vision Tool to determine the type.



Figure 44: UPC-A Composite Code.

There are also composite codes (see Figure 44) which consist of a 1-D and a stacked (PDF417 or MicroPDF417) code. The 1D portion could be any of the following: UPC-A, EAN-13, EAN-8, UPC-E, USS-128, RSS-14, RSS-14 Truncated, RSS-14 Stacked, RSS-14 Stacked Omnidirectional, RSS Limited, RSS Expanded, RSS Expanded Stacked and UCC/EAN-128. When reading a composite code, place the Vision Tool over the 1D part of the code. The Vision Tool will automatically search for the 2D component.

Since this type of Vision Tool consists of a line, it is susceptible to noise. To overcome situations where the noise causes codes to be misread, the Vision Tool contains an option to increase the scan lines. By increasing the scan lines, the user can rely on the output from most of the lines instead of all of them. This will allow some lines to misread the code with the Vision Tool still getting the desired result. This option basically creates a number of copies of the Vision Tool parallel to the original one at a user defined distance and groups all those Vision Tools into one.

When the position of the code is not consistent from image to image, the area-based version of the 1D Reader Vision Tool can be used. This Vision Tool will generate parallel scanning lines starting from the origin line progressively toward the opposite side until it can read the code. If the opposite side of the parallelogram is reached and the code was not read the Vision Tool will fail. This Vision Tool will handle translation of the code and some rotation, but it will not try to determine the orientation of the code.

An option is available to perform grading on USS-128, UPC/EAN, Interleaved 2 of 5, USS-39, Codabar and Code 93 barcodes. This Vision Tool can be used to grade the quality of the barcode being printed. The method follows ANSI/ISO standards and produces a full report for every barcode analyzed. For more information on this procedure, refer to the "Integration Notes" section of the DVT website or the Intellect help files. This procedure is very sensitive to illumination. Before using the Vision Tool for barcode grading, system reflectance calibration must be performed (see the Intellect help files for more information and review the section on Reflectance Calibration in this manual).

Two Dimensional Code Reader

In Intellect, there is one Vision Tool for all 2-D codes. This is a rectangular-shaped Vision Tool and the user will specify, under the "Options" tab the type of code that will be presented: DataMatrix, SnowFlake or Vericode (for Vericode, a license is needed).

The DataMatrix reader has the following features:

- Support for all ECC format DataMatrix codes, including ECC200. ECC 200 codes can be identified by the even number of rows and columns (result is a white pixel in the corner opposite to the intersection of the solid dark edges or vice versa).
- Automatic detection of symbol size starting at 10x10 (including rectangular code densities).

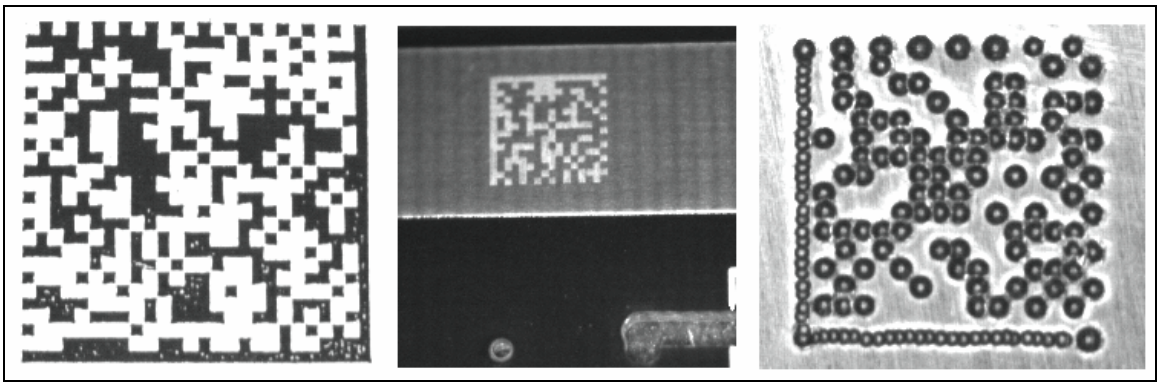


Figure 45: Different examples of DataMatrix applications: labels (left), printed circuit boards (center), and other metal parts (right).

The Vision Tool searches for a DataMatrix code inside its region of interest. That is, the code does not have to be consistently located in the same place every time. The Vision Tool offers robust draw-and-read performance. The reader works through training on a certain code. This training is performed automatically after the Vision Tool is drawn. After the reader has been trained, it collects information about the code and sets all trainable parameters: Symbol Type (e.g. ECC200), Symbol Size (in terms of number of cells), Module Type (dark modules or light modules), Mirror Image, Threshold Type and Cell Sample Size. For a detailed explanation on each of these parameters, consult the Intellect Help Files. The information learned about the code is then used in subsequent images to read each code. If a code cannot be initially read during the training phase, the user can manually set some of the known trainable parameters to help the Vision Tool find and read the code.

The Vision Tool searching process does not use an intensity-based threshold. The threshold indicated under the "Advanced" tab is only used once the code has been found in order to decode it. There are some parameters that can be set to specify certain expected characteristics about the code and that can help finding and reading codes that deviate from the learned code: Maximum Skew, Minimum Border Integrity, Minimum Quiet Zone Integrity and Maximum Axial Non-uniformity. Again, for a description of these parameters, refer to the Intellect Help Files. If the codes vary slightly in terms of size, a Scale Variation can be set under the "Advanced" tab.

The enhancements to the algorithms include substantial improvement in processing time. However, for those time-critical inspections, some parameters can be adjusted to further improve processing time. The

Minimum Gradient can be increased so that noise in the image is ignored during the searching process, especially if there is good contrast between the code itself and the background. Decreasing the region of interest may also improve the processing time.

This Vision Tool can be set up to simply read, decode and output the data, or to verify that a certain string is present in the code. For the second case, the Vision Tool will fail when the string is not matched. It also includes a digital relearn option. That is, if the string to be decoded changes, a signal could be sent to the Vision Sensor to relearn the matching string. The DataMatrix code can also be graded.

The 2-D reader Vision Tool can be set to read SnowFlake codes. Again, the code does not have to be consistently located in the same place every time. SnowFlake codes are designed for fast printing applications and dot matrix printers are generally used. Figure 46 shows two examples of such codes.

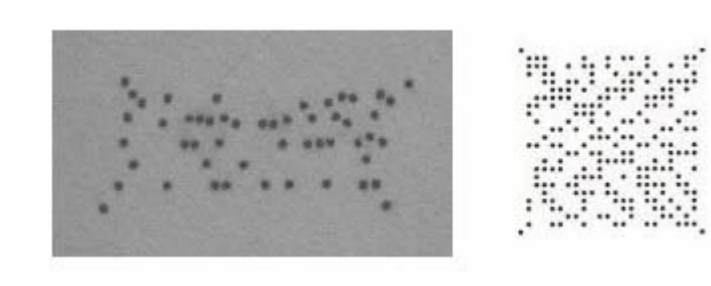


Figure 46: Examples of SnowFlake codes.

Several parameters can be set to improve the processing speed of the Vision Tool. The density, format, and type of error correction can be auto-detected. However, if these parameters are known, they should be manually specified to obtain faster reads. Other parameters that can be set include code symmetry, dot size and color (dark code on light background and vice versa). The SnowFlake Reader Vision Tool can be set to pass based on a match string.

Optical Character Recognition (OCR)

The DVT OCR (Optical Character Recognition) is a powerful industrial reading tool that provides robust character reading and verification capabilities. It is designed for reading lot or batch codes, serial numbers, labels, product IDs, and symbols in the industrial environment. It is not designed to be a page reader that reads scanned-in documents. Intellect's OCR Reader Vision Tool can read alpha numeric characters using either a user trainable font set or a fixed font set.

The trainable DVT OCR Vision Tool is based on feature extraction to learn characteristics about a character and then finds the closest matching character on subsequent inspections. Feature extraction computes the features of a graphical object and compares them to features of the trained or learned character set.

The fixed font OCR Reader uses a non-editable fixed font library that defines the character set. The fixed font reader is best used when the background is noisy but the characters are consistently printed.

The feature extraction process of the trainable OCR Vision Tool is a fast process that can handle scaled and rotated characters. Feature extraction is fully trainable, which means even symbols, foreign characters, or mixed font sets can be read. This is a very important feature; the trainable OCR Vision Tool is not limited to OCR fonts, which are fonts designed to be read by standard OCR readers. This tool can be used to read anything, the user only needs to train it first (i.e. teach the Vision Tool what needs to be read. In addition, it can be trained on a sequence of characters so that it can be used as an OCV Vision Tool. Instead of simply reading a code, it can also verify that a certain code is present. Unlike normal correlation, it can handle any type of character marking, such as dot matrix, laser etching, laser marking, ink jet, video jet, dot peening, etc.

The trainable OCR Vision Tool can be trained on any type of character with any type of font; however, some font styles work better than others. Special OCR fonts have been developed to make reading

easier by making characters with the same width (i.e., II and WW are II and WW) and by making similar characters as different as possible (i.e., 8 and B are 8 and B). Other factors that will help you achieve the most reliable and robust readings are:

- Achieve high contrast between the characters and the background.
- Make sure characters have good separation.
- Maintain a character height of between 20 and 35 pixels.
- Try to minimize noise in your image background.
- Keep characters as similar to the learned characters as possible.

This type of Vision Tool has two different types of threshold: intensity based and OCR. Intensity based threshold methods include the previously discussed types (static, dynamic, and color reference). The OCR thresholds are unique to OCR Vision Tools. This type of threshold is designed to maximize the extraction of the characters for more accurate readings. There are three options for the use of the OCR threshold:

- OCR Computed: Automatic intensity based threshold that makes an intelligent decision as to what is text and what is background, and sets the threshold level to better separate those two. It calculates a single threshold level for the entire area of the Vision Tool.
- OCR Linear: Same as OCR Computed but designed for uniform variations in light. It computes an individual threshold for the first and last characters and applies a linear variation of those values for the characters in between. The user needs to specify how many characters (blocks) the Vision Tool is supposed to read.
- OCR Nonlinear: Same as OCR Linear but for cases where the variation of the illumination is not uniform. The Vision Tool in this case computes one individual threshold for each character. This option requires more processing time than all others mentioned here.

This Vision Tool is very flexible; the user can customize it to the application. A great feature of this Vision Tool is the set of available shapes. The OCR Vision Tool comes in three different shapes: rectangle, parallelogram, and arc. The rectangle is used for text in standard position. The parallelogram is used for text that is inclined, like italics or text written at an angle. The arc is for text that follows a circular arc. Figure 47 shows four cases where the different shapes are being used.

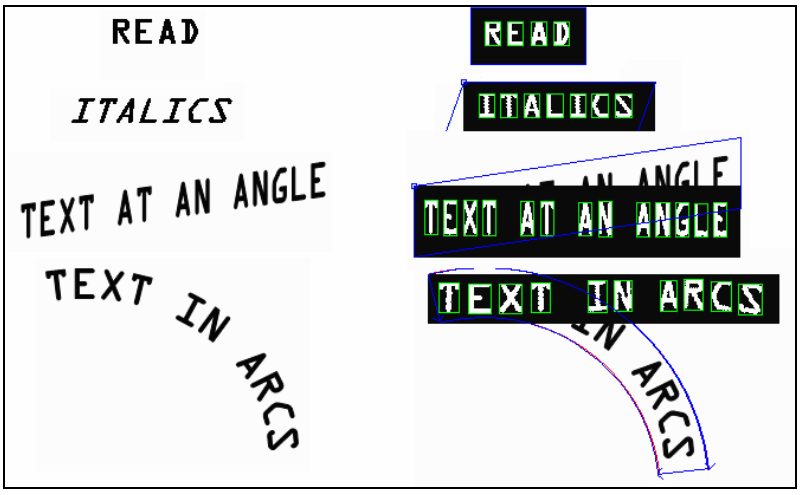


Figure 47: Uses of the different shapes of OCR Vision Tool. The top line uses a rectangle, the second and third lines use a parallelogram, and the last line uses an arc.

As the image shows, besides reading the letters in many positions, the Vision Tool offers image marking that straightens the text so the user can read it as well.

After the Vision Tool is drawn and the threshold is set, the user needs to isolate every character. That is, every character must have a red bounding box around it. When that is not the case, the user can change the extraction parameters (which limit the size of the characters in width and height), filter the image to eliminate noise, eliminate stray pixels, etc. The extraction section of the Vision Tool is a combination of very powerful processing options. As the user adds options the processing time for the Vision Tool will increase, so a general rule is to improve the image as much as possible to make the inspection more reliable and to minimize the processing time. Figure 48 shows an image containing non-conventional characters. This type of character is very difficult to read because of the deformation that it contains. The first image shows the text itself, and the second one the first attempt to isolate the characters.

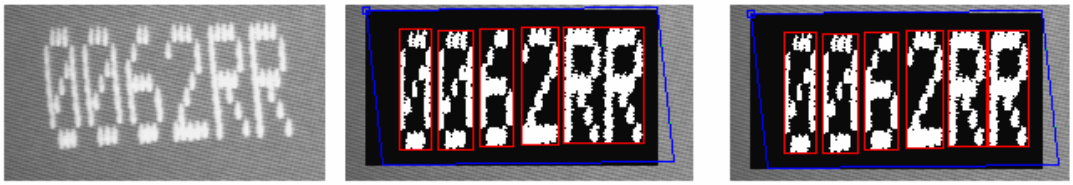


Figure 48: Use of the extraction tools to separate individual characters.

As the image shows, the last two letters share the same red bounding box. That means that the algorithm failed to correctly isolate every character. When the extraction parameters are being used (in this case maximum character width and advanced maximum character width) the Vision Tool makes a better decision as to where the characters end and they are correctly isolated.

The next step in the use is the training phase. The Vision Tool needs to be told what the extracted characters represent. This option is very powerful because anything can be trained, from user's handwriting to stroke characters. The only characters that are not trained are the spaces because the Vision Tool ignores them. Also, for applications where the speed of the process causes the characters to vary, several instances of each character could be trained to maximize reliability. As characters are trained, a font list is created and kept in the Vision Tool. This font list can be backed up to a PC for security purposes. If another system is to be reading the same labels, the font list could be loaded into the new system without the need of training all the characters again.

After training the characters, the user needs to establish the acceptance threshold that the characters being read need to match in order to be identified as one of the learned characters. When the Vision Tool has successfully learned a character or matched it to a learned one, the bounding box around the character changes from red to green. The user needs to specify a matching threshold that dictates the percentage of similarity that the characters need to have to be considered the same character. For sharp images with even illumination and very distinct characters, a high percentage could be set. As the image quality decreases, or variation between similar characters increases due to a poor printing process, the matching threshold must be reduced. In cases where some characters tend to be deformed but still need to be read, the ones that do not match the learned model should be trained as new characters. In such cases, the font list will increase for the Vision Tool will learn many instances of every character (for example 4 different uppercase A's). As the font list gets longer, the processing time for this Vision Tool increases due to the increased number of comparisons that it has to make. In order to overcome that increase in processing time, the Vision Tool offers two options: changes in resolution and auto fielding.

The Vision Tool has three different resolution levels: high, medium and low. They differ from each other in the number of features they extract from a character. High resolution mode extracts about 1600 features from the character, medium extracts about 800 and low extracts 200. The selection of medium or low resolution reduces the processing time at the expense of reliability in the identification of the characters. It should be mentioned that the different modes apply only to comparisons. That is, the characters are always learned in high resolution mode to make the comparisons more reliable. The second option, to reduce the processing time, increases reliability as well and is called character fielding. Character fielding allows the user to specify the type of character to look for in the different positions of the string to be read. When the application allows for it, the user can specify what type of character to

look for in every position of the string (code) to be read. For example, if a particular code consists of two upper case letters and two numbers, the user could specify that in the Vision Tool parameters. This will tell the algorithm to check the first two characters against all 26 upper case letters and the last two characters against all 10 numerical digits. This reduces the processing time by reducing the number of comparisons and makes it more reliable (ex: number 8 will not be confused with letter B because it is not being compared against it).

Another powerful feature of this Vision Tool is that it can search for characters. Once the Vision Tool learns some characters, it can be set to find one or all of the characters. This could be used for position reference for other Vision Tools (even for a second OCR Vision Tool). Before starting a search, the Vision Tool scans the list of trained objects to determine the maximum and minimum height and widths. From this, it creates a search window that is moved inside of the region of interest. When a "potential" edge of data is found, the Vision Tool moves through the font list, one object at a time, placing a "window" of the size of that trained object at the location of the "potential" edge. The edge is assumed to be the lower left hand corner of the window. As each window is placed, a read is made attempting to match whatever is in the window with any trained object. If more than one object matches, the object with the highest score is returned. If no object matches, the search window continues from where it was last, progressing bottom to top, left to right.

This Vision Tool can be set to pass or fail based on a string to be matched, one or more characters not being read, and a character being read but with a low score. The score represents how well the character matches the learned character. Scores range from 0 to 1000 to show how well the character is matched, a score of 1000 represents a perfect match.

Intro to Color Vision Tools: The RGB Color Space

In order to better understand the functionality of the Color Vision Tools the color space should be explained. As was mentioned before, color systems report four different values from each pixel. Those values refer to the color content and the intensity level. The color content is broken into three different coefficients to indicate the content of each basic color: Red, Green, and Blue. Any color can be created using the correct amount of Red, Green, and Blue, so by having the system report the contents of all three basic colors, any color can be represented in the system memory. The fourth parameter that is reported is the intensity level. This parameter would be the equivalent to the intensity as discussed for grayscale systems, and is computed from the values obtained for the basic colors according to certain weight associated to each one (based on the wavelength of the color). All four parameters (R for Red, G for Green, B for Blue and I for Intensity) range from 0 to 100, so color identification occurs in a three-dimensional space of R, G, and B coordinates whereas the intensity is an independent value. Figure 49 shows this color space. The X, Y, and Z axis are the R, G, and B components. A certain color could be defined by three coordinates; however, that will give a very specific color. In practice, colors are defined by ranges of R, G, and B. This defines a volume instead of a point as shown in the figure. Points inside that volume are of that color, points outside that volume are not of that color.

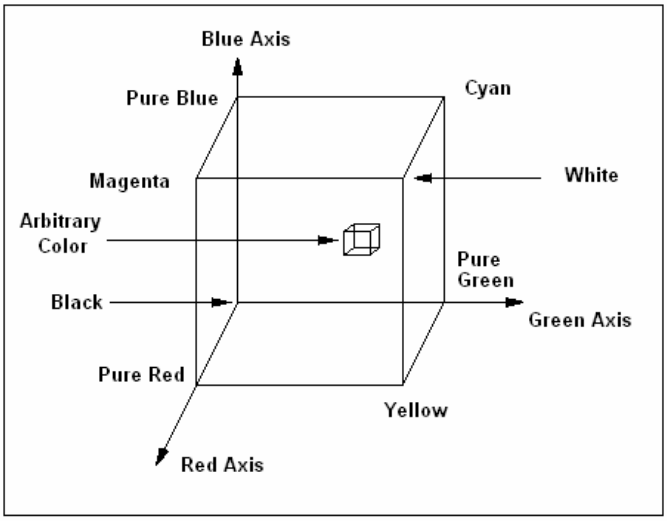


Figure 49: Three-dimensional color space defined by the basic colors Red, Green, and Blue.

This definition of the color space defines a new method for thresholding the images. The boundaries of the volume that defines the arbitrary color are the new threshold. Colors inside that volume are defined to be above threshold. Likewise, colors outside that volume are defined to be below threshold. As an example, if an image contains a number of objects of different colors, the user needs to learn the desired color. Once that happens, the user can use a color filter to extract only the desired colors. By doing this, the learned color will be the only color let through the filter. If the color of the object was learned thoroughly, the object will be shown through the filter because it is the specified color.

Pixel Counting Vision Tool (Color Vision Tool)

The Pixel Counting Vision Tool is the basic Color Vision Tool. This Vision Tool was introduced previous in the Counting section. However, we will explain this Vision Tool in further detail for when used on a color Vision Sensor. It is called Pixel Counting to emphasize what it does and to highlight the fact that its use is not limited to color systems. This Vision Tool learns a color (or many colors) and determines the number of pixels of that (those) color(s) that are found on subsequent images. As mentioned previously

in the counting section, if a grayscale system is used, the Vision Tool learns a shade of gray and uses it as if it was a color. There is not a concept of threshold in this Vision Tool. When the Vision Tool is drawn it learns the most dominant color within the learn region and sets a volume around it as shown in Figure 49. The space within which the color is defined can be changed using three parameters. Those parameters are:

- Sigma Factor: Changes the size of the space in all three axis in the 3D color space. Increasing it will include more (similar) colors in the volume. Decreasing it will make the color selection more specific.
- Matte Areas: Help the user select darker and lighter shades of the same color. In the 3D color space the matte line starts at coordinates 0, 0, 0 and passes through the selected color. The darker shades of the color are located between the color and black, the lighter shades are located on the other side of the matte line.
- Highlight Areas: Are used to include areas of the color that contain direct reflection of light. The areas affected by reflections tend to be white, so the highlight line goes from the selected color to white (coordinates 100, 100, 100).

Figure 50 illustrates the effects of the use of these parameters; the first diagram shows a learned color. The color occupies a limited volume in the RGB space. The image on the upper right corner shows the effect of increasing the sigma factor. The volume increases in all directions. At this time the color Vision Tool recognizes any color inside the new volume as the learned color. This makes the color selection more relaxed. If the user wanted to make it more specific, the sigma factor should be decreased to reduce the size of the original volume.

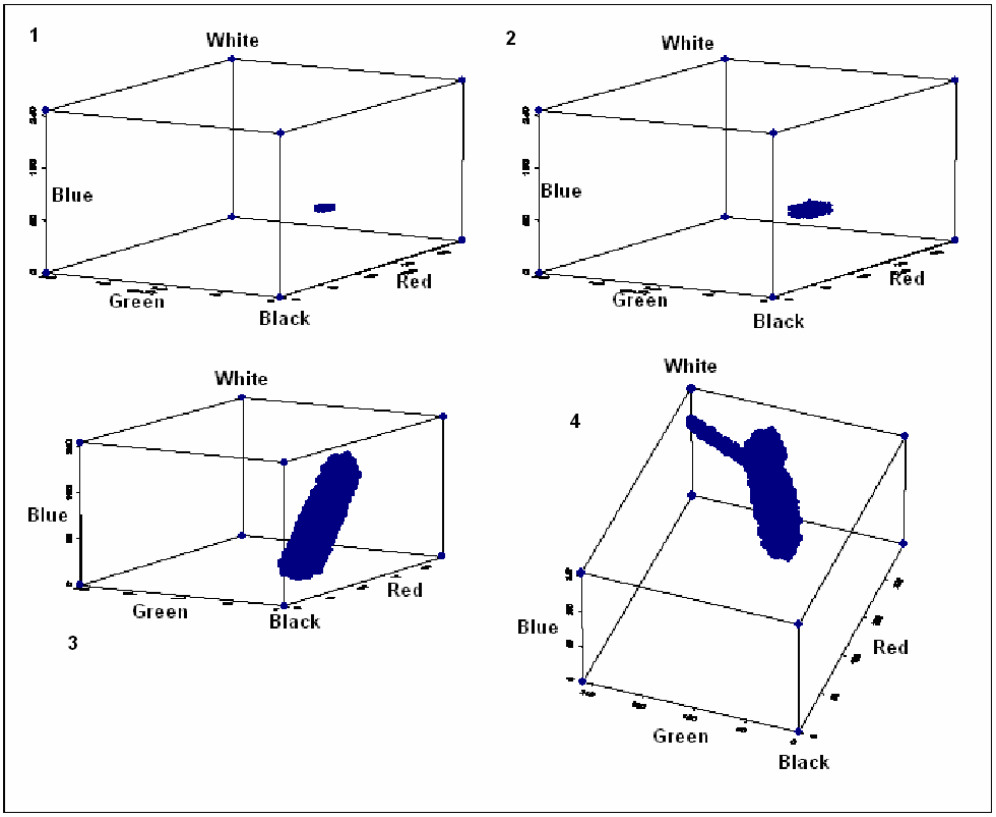


Figure 50: Three dimensional illustration of the effect of the color Vision Tool parameters. The first image shows the learned color, the second one the effect of increasing the sigma factor. The third one shows the effect of selecting the matte areas, both for darker and brighter shades of the color. The last image shows the effect of selecting the highlight areas.

The third diagram (lower left corner) shows the effect of using the matte areas. The volume is expanded towards black (coordinates 0, 0, 0) to pick up darker shades of the color or away from black to pick up brighter shades of the color. The user can select how much to extend the volume and in which direction (darker, brighter or both). Finally, the last diagram illustrates the use of highlight areas. This option is intended to include shades of the color that are caused by surface reflections. Those shades are between the original color and white (which would be a full reflection of the light). Increasing this factor expands the original volume towards white.

Figure 51 shows an example of the use of the sigma factor. The color Vision Tool learns only the light orange on the left end. The search region is over all the colors and image marking shows where the color is identified. For the example on the top row, it happens to be only over the learned color. This example uses a sigma factor of two. For the second example, the sigma factor was doubled. For a sigma factor of four, it identifies other colors as well. The last two examples represent sigma factors of seven and ten respectively. The Vision Tool is able to pick up different colors, colors that do not reside inside the original matte line, but as the sigma factor is expanded, those other colors are included in the volume that identifies the learned color.

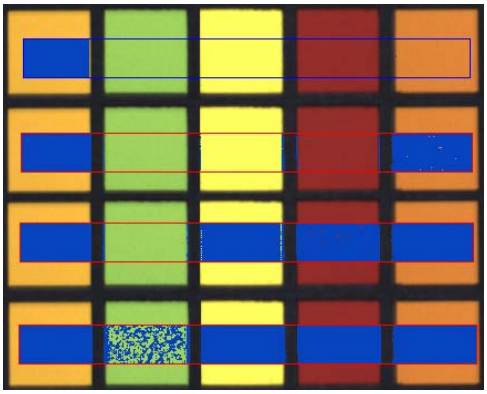


Figure 51: Illustration of the use of sigma factor to include other colors in the selection.

This Vision Tool can learn colors one by one. This procedure can be repeated to learn as many colors as necessary. The Vision Tool will keep a list of all the learned colors on the Manager tab.

There is another way to learn colors. The capabilities are under the "Multi Color" section in the "Training" tab. There are three choices: Learn a Merged Color, Remove and Add. Through the first option the user can learn a group of colors and merge them together all in one operation. The multiple remove and add operations will act on a group of colors previously learned and merged. The remove process will delete from the aforementioned group of colors that are in the region of interest of the Vision Tool at the time this action is executed. The add operation will do the opposite: it will add colors that were not present in the previously learned and merged group of colors. After executing any of these options, the user will end up with multiple colors grouped together under a single entry on the "Manager" tab.

Another feature that these Vision Tools present is the digital relearn. They can be programmed to expect an external signal (digital relearn), and when the signal is received, the Vision Tool executes one of several operations. These options go from a single color learning operation to automatic multi-color learn and merge processes. The digital relearn signal can be used to change the match color as well. The match color is the color against which every image is being compared to determine a PASS/FAIL result.

The uses for this Vision Tool are:

- To detect the correct color: The Vision Tool can be set to pass or fail based on color variation.
- To detect the right amount of color: The Vision Tool can be set to pass or fail based on changes in the amount of a certain color.

- To identify parts: The Vision Tool can be used to determine which part is in front of the Vision Sensor based on the "best match" feature, which tells the user which color from the list best matches its original size in pixels.
- To export colors: The Vision Tool can learn, merge and export the colors to other Vision Tools to be used as a threshold.

When the Color Pixel Counting Vision Tool is being used just to export the colors it learned, it might not be necessary for the system to keep processing the Vision Tool in every inspection. The user can disable the Vision Tool so that it can stay around but not consume processing time. When the option "Disable Inspection" (under the "Training" tab) is activated, the Vision Tool is only processed and inspected when a digital relearn signal is received and new colors are to be learned.

Color Monitoring

The Color Monitoring Vision Tool in the Flaw Detection group is another type of Color Vision Tool. It is a much more precise Vision Tool than the Pixel Counting Vision Tool. This Vision Tool can differentiate approximately 16.7 million colors, whereas the Color Pixel Counting Vision Tool can only discern about 65000 colors. As a result, the Vision Tool is used mostly to identify different colors when they are very similar. This Vision Tool reports the difference between colors. If many colors have been learned, the Vision Tool reports the color difference with respect to the closest color. The Vision Tool can be setup to work in the following color spaces:

- RGB is the default color space. Color difference is calculated as the Euclidian distance between two points. If Matte Line is selected, distances are measured to the matte line instead of the saved average color.
- Optical Density is the second option; in this case the color difference will be computed on a base 10 logarithmic scale. This color space is mostly used in the printing industry. RGB shows the reflectance of the measured sample, but Optical Density shows the absorption. The average Cyan, Magenta, and Yellow density is displayed. The reported color deviations represent the change in optical density of CMY components compared to the Closest Color. Color Difference is the maximum of DC, DM, and DY
- La*b*ΔE, this is CIE LAB. Please note that La*b* values are approximate. Standard formulas are used to compute these values.
- LC*h°ΔE_{cmc}, L is the Luminance or Intensity component of the sample. C* is the Croma or Saturation component and h° is the Hue angle. Color difference is ΔE_{cmc}.

During Inspection the average color of pixels under the Tool shape is found and displayed in the result panel. Depending on the color space used, the distance between this point and every color in the list is computed. The "Closest Color" is the one with the smallest distance. This distance is displayed in the results panel as Color Difference. Figure 52 illustrates how the distance is reported in the RGB space. The Vision Tool computes the Euclidian distance from the color found in the image to all the learned colors to determine the closest color. Once this color is found, the Vision Tool reports the color difference in each channel (R, G, and B) as well as the overall distance shown in the image.

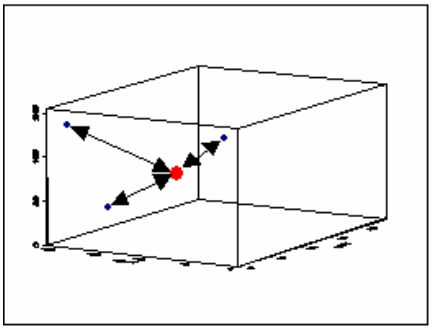


Figure 52: Illustration of the output (color difference) as the Euclidian distance in the RGB space. The Vision Tool reports the distance to the closest of the three colors that in this case have been learned.

An important consideration about this Vision Tool is that it does not mark the image. When the Vision Tool is drawn, all the pixels inside it are taken into account for the computation of the average color. The user is supposed to utilize this Vision Tool only over uniform areas of the color. For that reason image marking is not used (because it would mark every single pixel inside the Vision Tool every time).

This Vision Tool can be set to pass or fail based on five different criteria:

- Maximum Change in Color Data 1: Depending on the color space used it can be the maximum allowable change in Red reflectance, Cyan density, or Luminance.
- Maximum Change in Color Data 2: Depending on the color space used it can be the maximum allowable change in Green reflectance, Magenta density, a* or C*.
- Maximum Change in Color Data 3: Depending on the color space used it can be the maximum allowable change in Blue reflectance, Yellow density, b*, or h°.
- Maximum Color Difference: Is the maximum allowable distance between the sample color and the ClosestColor. The Color Difference is computed depending of the color space as explained above.
- Match Color Name: This allows the user to select one of the colors in the learned list. The name of the ClosestColor must match the selected color or the Vision Tool will fail.

Foreground Scripts (Script Vision Tools)

A foreground Script is a programmable tool that allows the user to write a short program with access to data from other Vision Tools. This program, which uses a very common syntax (similar to Java, C, and C++), is always part of a product and is executed when that product is used to inspect an image. That is why they are called Script Vision Tools. Script Vision Tools can:

- Be set to pass or fail based on certain conditions, except that the user defines those conditions. There is not a predefined set of rules because of the versatility of this type of Vision Tool.
- Access data from other Vision Tools and draw a conclusion (pass/fail) based on data from several Vision Tools.
- Perform mathematical calculations using built in functions or by letting the user program new functionality.
- Access registers in memory to interchange data between different products, with background scripts or even with external devices.

For more information, documentation about all the specific functions, and examples about Script Vision Tools please refer to the script documentation and Intellect help files.

Spectrograph Vision Tool

The Spectrograph Vision Tool is a Vision Tool designed to work with a special sensor in the DVT Vision System family: the DVT Spectral Camera. Figure 53 shows the functionality of this hardware device. It consists of two stages of image acquisition. Stage 1 is a regular lens that has to be used to correctly focus on the part. The focusing lens can be removed for applications where the camera is inspecting a single color within the FOV. Stage 2 consists of another optical device that takes a vertical line from the image focused by the lens and spreads the spectrum of that line over the CCD. When this image is projected onto the CCD, it consists of a grayscale image with no defined shapes in it. That is because the spectrum consists of different levels of intensity at different locations in the image. The X axis of the image becomes the Spectral axis, that is, if the external optical device is correctly placed, blue tones should be on the low X coordinates and red tones should be on the high X coordinates. Likewise, the Y axis represents the Spatial axis. This axis is the same as for a regular image but applied to a very narrow opening. In other words, a change in X coordinates would represent a change in wavelength, whereas a change in Y coordinates would represent a change in position.

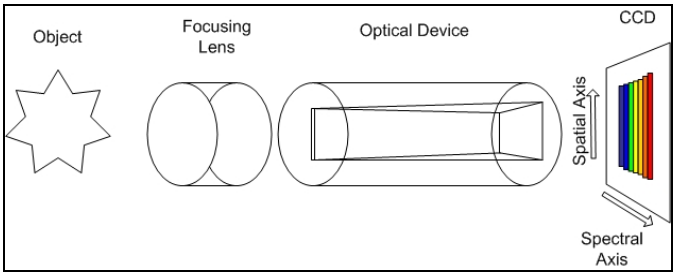


Figure 53: Functionality of the DVT Spectral Sensor.

Figure 54 shows an image obtained with the DVT Spectral Sensor and the pixel graph that the Spectrograph Vision Tool obtains from it. As we analyze the image, we can see that it consists of different levels of intensity. The Spectrograph Vision Tool is an area Vision Tool that is only available as a rectangle.

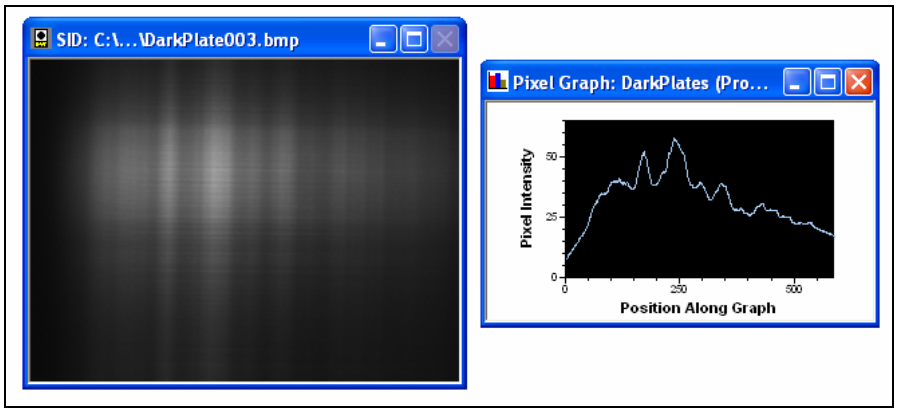


Figure 54: Image obtained with a DVT Spectral Sensor and pixel graph of the Spectrograph Vision Tool.

The functionality of the Vision Tool is shown in Figure 55. The squares represent the pixels and the text inside them represents the intensity level.

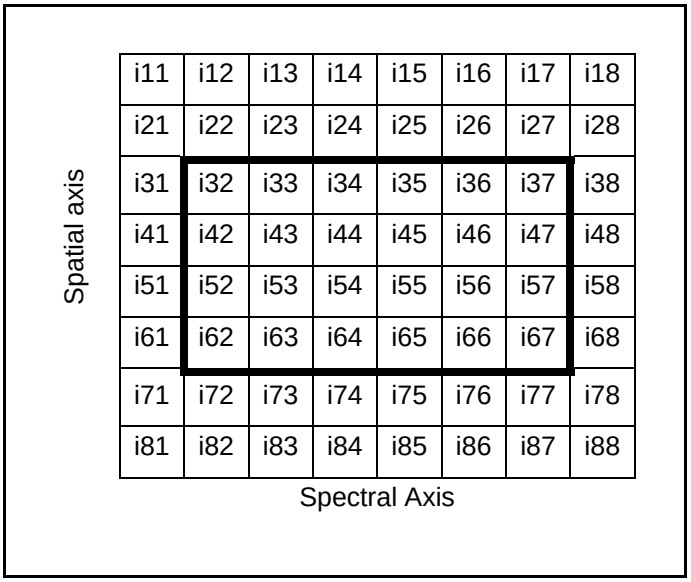


Figure 55: Pixel grid with intensity levels and Spectrograph Vision Tool interpreting the data.

Pixels are arranged in matrix style so we use matrix naming convention. The intensity level at the pixel with coordinates (1,1) is given by i11, the one for the pixel with coordinates (3,5) is given by i35, and so on. The Vision Tool is represented by the dark rectangle inside the pixel grid. In this case the Vision Tool extends from pixel (3,2) to pixel (6,7). From the data inside this rectangle the Vision Tool must compute a spectrum. In order to compute a spectrum the Vision Tool must use only one data point for every wavelength, and since the X axis is the spectral axis, it must use one single data point from every column of pixels inside the Vision Tool. the Vision Tool averages the pixels in each column to obtain a single data point

The DVT Spectral Camera will come calibrated in nanometers and the user can recalibrate at any time. This means that peak reflection locations will be specified in this unit rather than just image coordinates. The user can also perform binning operations in both the spectral and spatial axes. Binning can be used in applications where the brightness has to be increased without increasing the exposure time. Since the DVT Spectral Sensor will be calibrated, it will be possible to compute absolute color data for different color spaces. After a Dark Image Calibration is done, the user can select the option to subtract the black image from the spectrum being analyzed by the Spectrograph Vision Tool.

The functionality of this Vision Tool is similar to that of color Vision Tools. The user can learn different spectrums and compare the observed spectrum with the learned ones. In order to maximize accuracy in the comparison, the user can scale the spectrums to the peak values.

This Vision Tool can be set to pass or fail based on maximum spectrum difference, maximum change in location of peak wavelength, and maximum change in peak reflection using a learned spectrum for comparison.

Chapter 4 – Inspection Application Guidelines

This chapter provides major guidelines to approach different inspections. Users should be familiar with the different types of Vision Tools available in order to understand the concepts and issues explained here.

Dealing with Colors

The main uses of the Pixel Counting Color Vision Tool (Color Vision Tool) are to detect defects and to verify that correct parts are used in an application. Defect detection can be performed by learning the "good colors" of a part and detecting any color not in the "good colors" list in subsequent parts. If the type of defect is known, the "bad colors" can be learned instead and the images can be analyzed by checking for the presence of those colors only. Part presence can be detected by learning the colors of a number of parts and verifying that the correct part is being used by inspecting subsequent images and comparing the colors present to the desired color. Another way in which colors are used is to export them to other Vision Tools. When that is done and another Vision Tool references the color, areas of the imported color become light areas (referred to as above threshold for grayscale systems), and areas of other colors become dark areas (referred to as below threshold for grayscale systems). There are two ways to learn colors using the Pixel Counting Vision Tool: single color learning and multi color learning. The process for single color learning is illustrated in Figure 56.

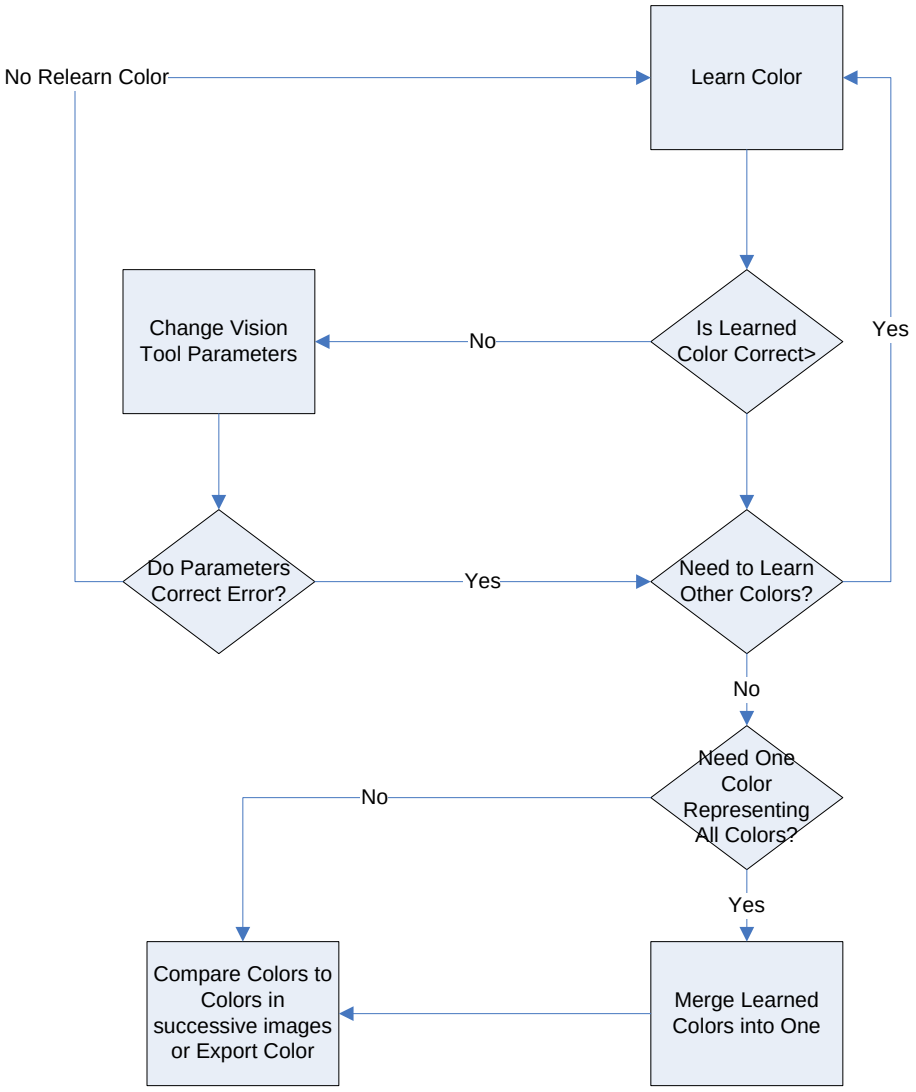


Figure 56: Process for single color learning.

As the process in the figure illustrates, single color learning does not mean the use of a single color but the learning of one color at a time. By following this process, the user will end up with a list of colors (or

a single color representing the list) that describe a part or a set of parts and can be used for future inspections.

For this process, only one color can be seen by the Vision Tool when the colors are being learned. Any attempt to learn a color when two or more colors are seen by (are inside of) the Vision Tool will result in incorrect colors being learned.

The second use of the Vision Tool is the multi color learning. This process involves learning multiple colors at the same time. The process is illustrated in Figure 57.

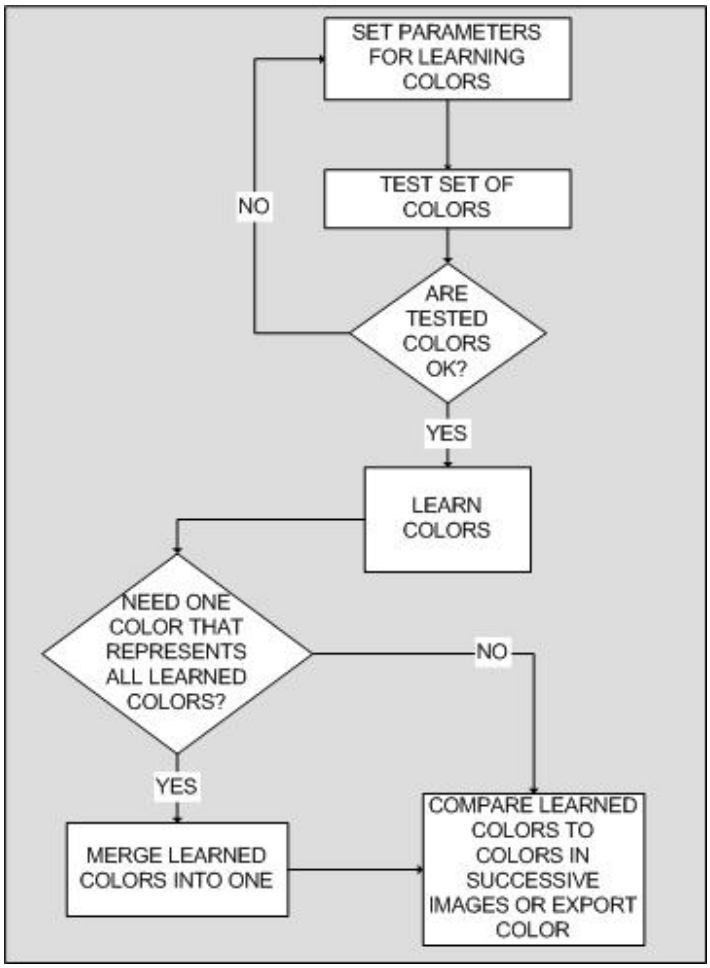


Figure 57: Process for multi color learning.

This process is used to learn the colors of a complex part- one that is not of a well-defined color. The Vision Tool is drawn over the entire area (so all the colors are inside the Vision Tool) and the process begins by setting the parameters. The parameters include the minimum number of pixels that each color must have to be included in the list (Cluster Size) and the maximum number of colors to learn from the part (Maximum Cluster Count). Based on those parameters the image is tested to get image marking showing which colors are learned. Only after the desired colors are marked should the user learn the colors. Since this method learns multiple colors at the same time, the list of learned colors replaces any colors previously learned. For more information about color parameters see Pixel Counting Vision Tool (Color Vision Tool).

Dealing with Part Movement

In real world applications, parts to be inspected are not consistently placed in the same position for every inspection. There are many parameters that affect this such as the use of proximity switches to indicate

presence/absence of the part. Those switches will only tell whether the part is present or not. For more precise information about the location of the part, proximity switches are not very reliable. Intellect offers a number of tools to help the user locate the part in the image and to create a reference that the inspection Vision Tools can use to track the movement of the part. This set of tools consists mainly of Vision Tools in the Positioning group, however Vision Tools in the Measurement and Counting Groups can also be used for positioning. These Vision Tools are used to calculate part movement. The user needs to create a reference from the inspection Vision Tools to the positioning Vision Tools so every Vision Tool inspects the correct area of the image every time. Figure 58 illustrates how to track the part. The positioning Vision Tool locates the part and calculates the shift in position (with respect to the original position).

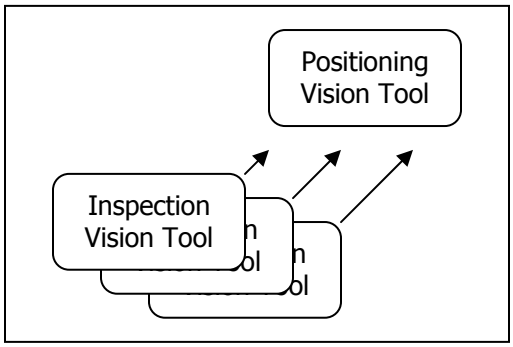


Figure 58: position reference mechanism.

The inspection Vision Tools then reference the positioning Vision Tool to find out where the inspection is to take place. If the application allows for the use of the most powerful Area Positioning Vision Tools (Blob and ObjectLocate or PatternMatch Vision Tools) just one Vision Tool can produce all the necessary position references. However, some of the fastest positioning Vision Tools can only track one type of movement. Unfortunately, in most cases a part translates in both horizontal and vertical directions and rotates. When that happens and the Vision Tools used do not provide complete position feedback, the user needs to create a chain of references. Vision Tools may need to find the changes in the location of the part in both X and Y directions and then determine the rotation of it. In those cases, the chain of references is extended to multiple levels. Figure 59 shows consecutive images taken from an actual inspection. As the images show, the part moved. If the Vision Tools are to inspect it, positioning Vision Tools need to find it in the image. In this case, the part shifted up, left, and rotated counterclockwise.

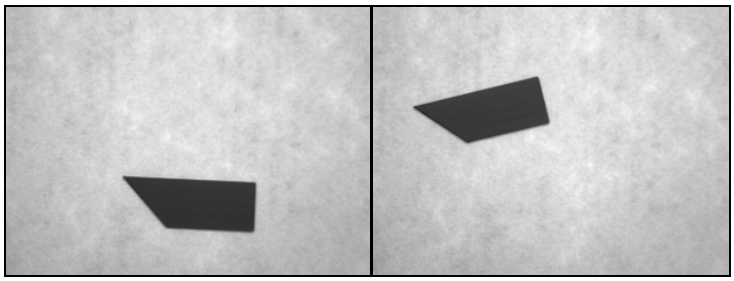


Figure 59: Consecutive images of parts to be inspected. The image on the left shows a part in the original position whereas the one on the right shows why positioning Vision Tools are needed.

The procedure in this case would be to find the vertical displacement (the one that changed the most) first. Then the horizontal displacement, and only then determine the rotation. The diagram in Figure 60 illustrates the process.

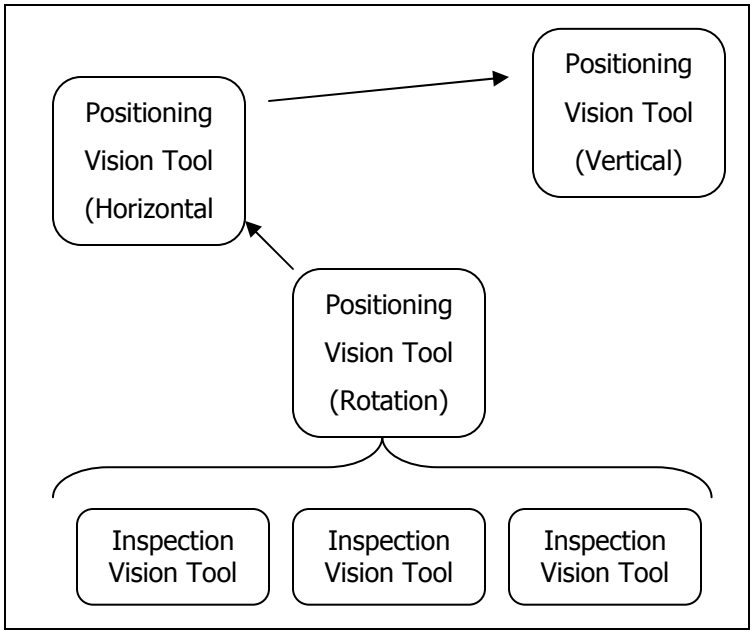


Figure 60: Method for passing references.

The Vision Tool detecting vertical movement finds the change in position in the "Y" direction. Then, the Vision Tool tracking changes in the horizontal position uses that information to look for the part in the upper region of the image. Once it finds the part it automatically records the change it detected plus the change obtained from the other Vision Tool and passes both as a position reference to the Vision Tool detecting rotation. This Vision Tool shifts up and to the left to analyze the rotation in the right location and adds it to the chain of position references. When the inspection Vision Tools request information from this one, they obtain the changes in position generated by all three positioning Vision Tools. This method simplifies the setup process because position references are automatically concatenated and the inspection Vision Tools need only reference one of the positioning Vision Tools.

As mentioned before, just one Area Positioning Vision Tool like the Blob, ObjectLocate or PatternMatch Vision Tools could have been used, but they would have consumed more processing time. The Blob, Object and Pattern finding algorithms are area based algorithms, that is, they scan an entire area looking for a shape. This process takes more processing time than a line scan like the one performed by line Translation Vision Tools.

Presence/Absence Inspections

Presence/Absence inspections involve inspecting a part for specific features to make sure the feature is present or absent. The locations of the features of interest in a Presence/Absence inspection are generally known. Many of the Vision Tools within Intellect can be used to inspect for the presence or absence of features of an object. Counting Vision Tools are commonly used for presence/absence, but Measurement, Identification, Flaw Detection and Position Vision Tools can also be used to determine the presence or absence of a part or feature. Each one of these Vision Tools has different advantages and disadvantages. Line based Vision Tools like the FeatureCounting and EdgeCounting are the ones that take the least processing time, but they are more sensitive to noise. Blob, Object, and Pattern Counting take more processing time but they offer a number of features to filter out noise. In every case, image contrast plays a very important role. The objective of the physical setup of the application should be to maximize contrast between features to be inspected and the background. Figure 61 shows an example where the use of a different lens, correction of working distance, and use of appropriate light source increases the contrast between the part and the background.

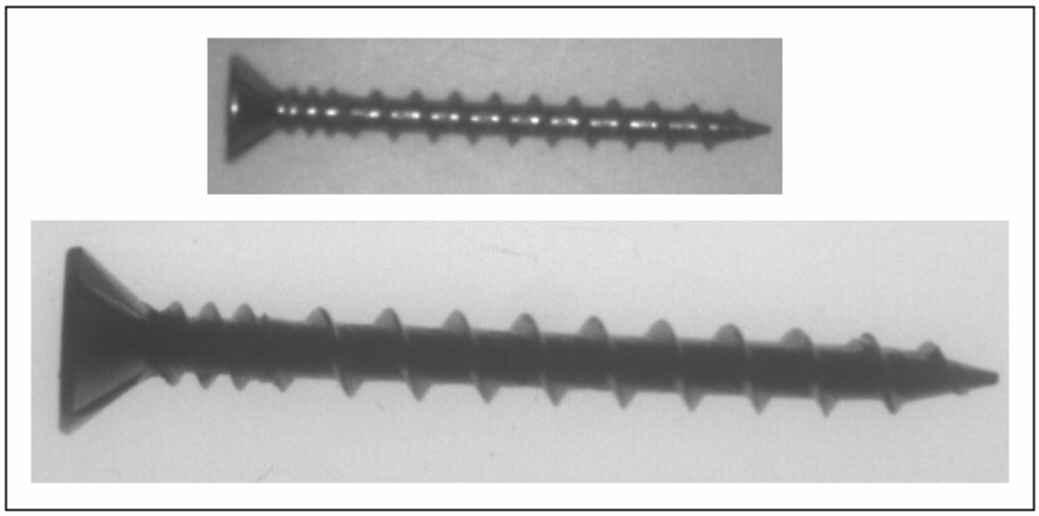


Figure 61: Importance of physical setup of the application to maximize contrast between part and background.

The image on the bottom is a better image, due to the higher contrast between part and background, the absence of reflections (no pixels are saturated), and sharper.

Flaw Detection Inspections

Flaw Detection inspections typically involve looking for a general flaw on the surface of a part where the location of the flaw is unknown. Typical Vision Tools used for flaw detection are located in the Flaw Detection Vision Tool group, however other Vision Tools can be used for inspecting for flaws. The choice of Vision Tool depends on the application. Defect Detection can be used if the part contains a certain pattern that can be learned and searched for in subsequent images. This Vision Tool takes a lot of processing time and system memory, since the Vision Tool needs to keep the learned model in memory. Color Monitoring can be used when a specific color is desired and the flaw occurs when the color does not match. When the flaw can be isolated using a threshold level, the other tools can be used to find it. Pixel Counting and Feature Counting Vision Tools are the Vision Tool that take the least processing time.

Counting Inspections

Counting applications require parts or features of a part to be counted. The Vision Tools in the Counting Vision Tool group are typically used for counting applications. The Edge, Feature, and Pattern Counting algorithms in the Count along a Line Vision Tool should be used to identify features along a line, arc, or another predefined shape. These Vision Tools scan along that line counting the edges or features found. They take little processing time, but they require that the pattern in which the features are arranged be known. Blob Counting should be used when counting parts that change the position and that do not touch each other. The Blob Vision Tools will count two parts as one if the blobs touch one another. If the parts are separated, this type of Vision Tool offers a very reliable solution at a relatively high speed. When parts can be touching or even overlapping, the ObjectLocate algorithm in the Count in Area Vision Tool should be used since it learns the shape of the part and will find it even if the entire shape is not visible. In this case the Vision Tool will report that there is a part when a certain number of features learned from the part are found in the image. The user can select the acceptance level to discard parts that are barely visible or severely damaged.

Measurement Inspections

Measurement applications involve measuring diameters, widths, height, and other key measurements of an object. Usually, a combination of Measurement Vision Tools solves the problems. When more operations are needed, scripts can be used. For instance, scripts or the Statistic Vision Tool offer the flexibility needed to maintain running averages between inspections.

Lighting

Backlighting should be used for measurement applications whenever possible. This technique consists of placing the part over a plane of light to highlight its contour. The image produced by such arrangement simplifies the job for measurement tools, and makes them more precise too. When backlight is used on objects with curved surfaces, the light should be masked or collimated to obtain better results. Figure 62 shows examples of the use of masks and collimators. The image on the left uses standard backlight. Since the object being inspected has curved sides, the light produced at the far ends of the light source bounces on the upper regions of the part and is directed towards the Vision Sensor. This causes the Vision Sensor to see fuzzy edges making the measurements less reliable. When this happens, edges tend to be defined by transitions of as many as 10-20 pixels. The second example shows the effect of masking the backlight. Masking simply means covering the areas not used, or the areas farther away from the edges of the part. This will make the edges of the part appear sharper (transitions of 3-4 pixels) thus making the measurements more precise. When there is part movement and the light cannot be masked, a collimated light should be used; this is shown in the third image. The collimator filters the light so that only rays perpendicular to the surface of the light source get to the Vision Sensor causing a similar effect as masking the light.

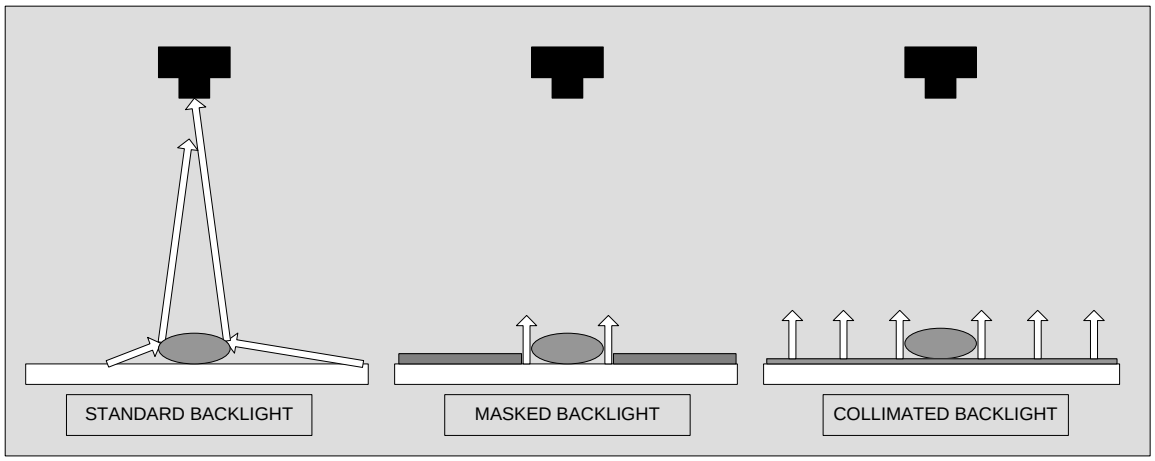


Figure 62: Examples of the use of masks or collimators to enhance part edges when using backlight.

Another important consideration for obtaining precise measurements is to avoid altering the intensity levels in the image. There should be no saturated pixels (pixels reporting 100% intensity level).

Lensing

The type of lens used for measurement applications should minimize lens (or barrel) distortion. The best choice is to use telecentric lenses. These lenses minimize distortion and perspective error. Using a telecentric lens the part size and shape should not vary if it moves within the FOV or even farther and closer to the lens. This type of lens can get very expensive as the part size increases. An inexpensive alternative consists of zero distortion lenses. Most lens manufacturers carry a number of this type of lenses.

Positioning Robotics Applications

Applications requiring the position of an object to be obtained involve Vision Tools that can generate the location of a part feature and conversion from Image coordinates to Robot/Real World coordinates. Vision Tools used for these applications are generally Measurement/Math Vision Tools with a Coordinate Transformation Vision Tool for converting to real world coordinates. Object, Blob and Pattern locating using an Area Positioning Vision Tool can also be used if general position location is desired. This Vision

Tool requires more processing time because they scan entire areas searching for objects. A Coordinate Transformation can be used both to change pixel coordinates to robot coordinates and to correct for lens distortion (both require previous calibration). The second correction can be minimized by using a better type of lens.

Setting up a Coordinate Transformation

A coordinate transformation establishes a relationship between the Vision Sensor image pixels and the plane of the real world that the Vision Sensor is imaging. This allows positions and measurements to be reported in the robot space. A coordinate transformation is established at the system level and can be used by any products in the Vision Sensor. There are three methods to generate a coordinate transformation. Standard DVT Grid, Custom Grid, and Script Based. Using a Standard DVT Grid allows for a quick and easy coordinate transformation by placing a special calibration grid in the field of view and performing a calibration. A custom grid allows the user to use a grid pattern with a set dot pitch. A script calibration allows the user to define pixel point to real world point pairs in script. It is recommended to use the standard grids when possible. These grids are available on the DVT website in the download section.

The steps to create and calibrate a Coordinate Transformation are explained below:

- Add a new Coordinate System in the System Menu section.
- Select the Grid Type.
- Place the appropriate grid in the Field of View
- Click right Mouse button on the Transformation in the System Explorer and select Calibrate Coordinate System.
- Open the Product Manager in the Product menu and Select the desired product to use the Coordinate system. In the properties window of the product, set the Coordinate Transformation for the product.

All the measurements and positions will now be displayed in the appropriate real world system. You can apply a Coordinate Offset and scaling factor to the Coordinate system to alter the reported position and measurement values.

Color Sorting / Identification Applications

Color Sorting applications generally involve monitoring the color(s) of an object to determine which object it is. The Color Identification Vision Tool can be used to determine the part based on the distribution of colors in the part. Using this Vision Tool a multicolor model can be used. For very similar colors, which are difficult to differentiate with regular color Sensors, the DVT Spectral Sensor should be used in combination with the Spectrograph Vision Tool. Basically, for gross differences the Pixel Counting Vision Tool should be used. For parts that offer very little color difference and when the Pixel Counting Vision Tool does not offer the reliability needed, a Color Monitoring Vision Tool should be used. This Vision Tool requires that the area to be analyzed be of a uniform color, no color variations are allowed. In cases where the color difference is visually hard to detect, the DVT Spectral Sensor should be used to compare the entire spectrum of the parts and make a more reliable decision. The use of the DVT Spectral Sensor requires that the part to be analyzed be consistently placed because the positioning Vision Tools cannot not used by the DVT Spectral Sensor.

Part Identification

Identifying a part is a very broad type of application that generally entails defining differences between the parts and identifying those using Vision Tools. There are several different techniques for achieving part identification and they depend mostly on the part being inspected. The Identification Vision Tool Group contains Vision Tools that can learn multiple models using either color properties of the parts or

blob properties. Other Vision Tools that can be used by themselves to identify parts are Blob Locate, Object Locate, Color (Pixel Counting), Reader, Pattern Locating Match Vision Tools. Blob Vision Tools should be used when the parts can be identified based on general geometric features (Area, Radius, Eccentricity, Compactness, etc.). When a more precise identification needs to be done, Object Locate should be used at the expense of some extra processing time. Pixel Counting Vision Tools should be used when the difference between parts is based on color. Finally, Reader Vision Tools should be used when the difference is in a barcode, DataMatrix, or a label with characters. In those cases, the user needs a Vision Tool to read the data from the label or part in order to determine which part it is.

Lot Code and Data Code Applications

Lot Code and Data Code Reading can be accomplished with any one of the Reader Vision Tools. The OCR (Optical Character Recognition) Vision Tool is used to read human readable characters. The 2-D and 1-D Readers are used to read information symbols printed on the product.

OCR

The OCR Vision Tool can be trained to read any character (not just letters or numbers). Usually this type of application requires positioning. That is, the label is not located always in the same location. This problem requires more Vision Tools to provide a position reference. As mentioned before, the OCR Vision Tool itself can be used for position reference. The user simply needs to create an OCR Vision Tool to locate the first character and another Vision Tool to read the code. This option of finding the first character makes the positioning very independent of noise, since the Vision Tool will be looking for a specific learned character, not an edge or a feature.

1D Reader

The 1D Reader Vision Tool consists of two line-based tools (arc and straight) and an area based Vision Tool. The line-based versions of the Vision Tool scan along a line trying to read a certain bar code. The user has a few options to speed up the process or to make it very thorough. When the application consists of the same type of code every time, the user can set the Vision Tool to look for that specific code. This will reduce the processing time because the Vision Tool does not need to attempt to decode the data using other methods. When the application determines that different types of barcode are to be used, the user must select the option to remain in auto detect. In this case the Vision Tool not only reads the code but it also determines what type of code it is. This option requires more processing time.

The parallelogram version of the 1D Reader Vision Tool will help in cases where the position of the code changes from image to image, handling translation and some amount of rotation.

2D Reader

There DataMatrix reader is a searching Vision Tool. The DataMatrix Vision Tool searches for the code in a predefined area. Even if the code rotates, the Vision Tool will find it and read it. This option takes more processing time because the Vision Tool needs to search for the code in what usually is a noisy image (typically the code is surrounded by text). In order to speed up the process, the user should provide the Vision Tool with as many parameters as possible about the type of code to search for, so the searching process is faster.

The SnowFlake reader in Intellect has a rectangular shape and is a searching reader.

Chapter 5 – DVT Vision Sensor Integration

This chapter discusses the different ways in which the user can send data to the DVT Vision Sensor or get data out of it. The most common method is with the digital I/O lines, which were described under System Parameters. This chapter explains how to implement Modbus transfers, how to set up DataLink, how to set up the built-in drivers, and how to enable/disable different operational modes. It also mentions the types of communications that DVT Vision Sensors support.

Transferring Data from a DVT Vision Sensor

The most important factor to consider before setting up a data transfer is the timing. Information about an inspection needs to arrive in a certain way for the receiving system to interpret it correctly. There are two ways to transfer data: synchronous and asynchronous.

Synchronous transfers of data happen after every inspection. The user sets up the data to be transferred and that data is sent out after every inspection. This ensures the receiving system that new data is transferred as soon as it becomes available. The methods of communication that support synchronous transfers are the digital I/O (discussed under system parameters), and DataLink.

Asynchronous transfers of data happen at a certain rate. In most cases, the user selects the rate and blocks of data are continuously transferred whether they contain new data or the same data that was already transferred. An example of this method of data transfer is the Modbus master transfer available from Intellect.

It should be mentioned that because of their flexibility, scripts allow for both types of data transfers, but in this case the user needs to implement the transfers. For more information about scripts see the script documentation and help files.

DataLink

DataLink is a built-in tool used to send data out of the system and even receive any terminal commands from another device. This tool is Product-specific, that is, every product has its own DataLink that can be configured depending on the inspection and the Vision Tools being used. DataLink consists of a number of ASCII strings that are created based on information from the Vision Tools. By default, the datalink string is sent out after each inspection; however the user can define a set of rules under which those strings are to be sent out. Those rules are created using logical operators (and, or) acting on Vision Tool results and associating them with a User output. For example, if the positioning Vision Tool fails a certain string can be sent out, but if this one passes, DataLink could look at the result from another Vision Tool to determine which string (if any) to send out.

Once the conditions are indicated, the strings need to be created. This is a simple point-and-click process that gives the user access to data from every Vision Tool in the Product plus the possibility of typing some extra characters. Figure 63 shows a screen capture of DataLink string editor.

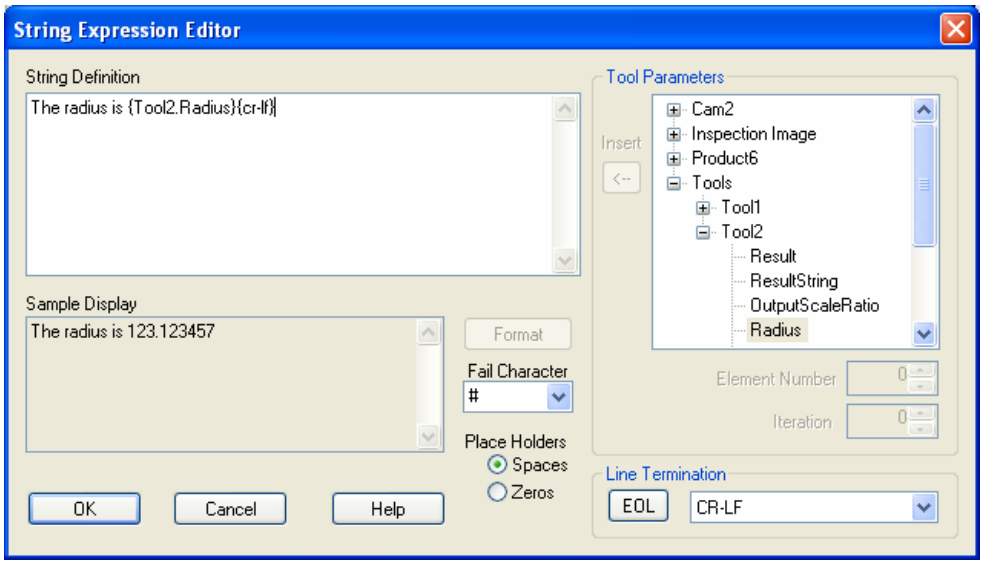


Figure 63: String setup for DataLink.

The last section of the string is the "end of line" character, which is available from the menu shown in the lower right corner. In this case {cr-lf} was selected which gives a carriage return followed by a line feed.

Depending on the output from every Vision Tool, one of the selected messages (Strings) will be sent out or, in some cases, no string will be sent out (ex: if all the Vision Tools pass). This will give the user the tools to determine where the flaw is in the process and allow him to take immediate action. The fact that every string is fully configurable gives the user the flexibility needed to implement an effective inspection. In order to read the data sent out from DataLink, the user needs to establish an Ethernet connection using TCP protocol to port 3247.

DataLink output can also be sent in XML format. XML is becoming the standard for specifying the actual meaning of data. In this case the names of the tags and overall format are hard-coded. Additional meta-data is provided beyond what DataLink's regular format provides so as to make parsing easier. For example, the parameter's name and format (int, float, etc.) is given, as well as the associated Product name. To enable this new output format the user should select the "XML Output" for the output type in the datalink properties page.

Modbus Transfers

A very efficient way to share data among DVT Vision Sensors or between a Vision Sensor and an external device is to perform a Modbus transfer. Modbus transfers, as implemented in the Intellect user interface, take a number of registers from one system and copy those registers into another system. Using this simple procedure, one Vision Sensor can make a decision based on the result from other Vision Sensors or gather data from other Vision Sensors and send it out to an external device. In a Modbus network we have master devices and slave devices. Vision Sensors can be enabled to be a Modbus slave, which will make the system expect a connection on port 502 (Modbus standard). The Modbus Master initiates the connection and decides the type of operation to perform (write data to the slave or read data from it). Figure 64 shows two possible configurations for sharing data using Modbus transfers. The first configuration uses Modbus transfers to exchange data between Vision Sensors. There are four Vision Sensors in the DVT network: three of them are being used as Modbus slaves and the fourth one acts as the master. This Vision Sensor queries the slave Vision Sensors for data and, based on the data from all three slaves and its own data, draws a conclusion. The final output of the inspection is made available to an external device using either Modbus or some other type of communication. It could even be a single I/O line indicating a PASS or FAIL result. The second configuration shows a slightly different setup. In this case, an external device is acting as a master and the Vision Sensors performing inspections are acting as slaves. The external device has direct access to the internal memory of the Vision Sensors so it can easily query the individual inspection results from each.

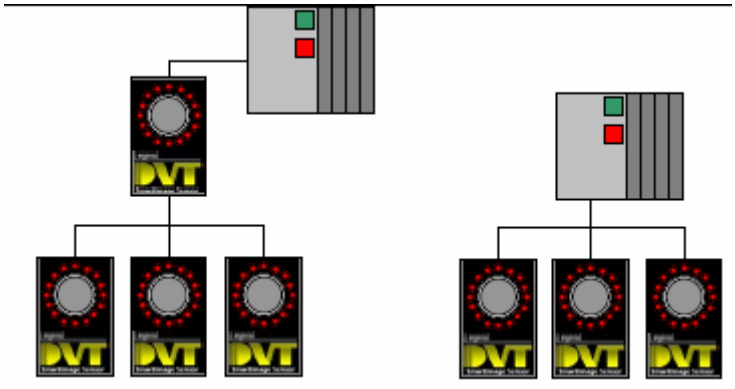


Figure 64: Two different configurations to share data among multiple devices.

In order to make a Vision Sensor behave as a master, a Modbus master transfer must be created from Intellect. This transfer will indicate the slave IP Address, the operation to be performed (read/write), the block of registers to use, and the polling rate for the transfer. Note: by default the master will transfer

data both at the specified polling rate and at the end of each inspection. In order to specify the block of registers to transfer, the user must be aware of the difference register sizes for the different data types used by the Vision Sensor.

Modbus registers are 16-bit registers. Therefore, it is important to take into account is the size of the different DVT data types. This becomes a very important issue when trying to transfer data. Different data types occupy different numbers of registers in memory. If a certain value is saved to memory using a script tool and that particular piece of data needs to be transferred to another system, the user has to know the exact number of registers that the piece of data occupies in memory. Table 4 shows the size that every different data type occupies in memory. For example, if a measurement tool outputs a float, that number will occupy 32 bits. So, according to the size of the Modbus registers (explained above) this requires the transfer of 2 Modbus registers.

Table 9: Size of the different data types in bits and Modbus registers.

Name	*Size (bits)*	*Size (Modbus Registers)*
Short	16	1
Integer	32	2
Long	64	4
Float	32	2
Double	64	4
String	8 per char +1	1 per 2 char +1

For more information about Modbus protocol and Modbus transfers see the DVT Integration Notes section of the DVT website.

If the transfer of data needs a more explicit process, Modbus transfers might not be the best approach. In those cases a background script should be used. From a background script, the user can create a Modbus transfer object which can perform a number of standard Modbus functions as specified in the open Modbus protocol. For more information about this see the script documentation.

Ethernet Terminal Controller (ETC)

The Ethernet Terminal Controller (ETC) allows the user to set up different Ethernet configurations for communication using the TCP protocol. This converts the Vision Sensor into a TCP server or client. Depending on the application, the user can select the Vision Sensor to be a client (connect to another device) or a Server (expect a connection from other devices). In both cases, the connection will be established using a user-defined port. Using this connection the following drivers can be used to communicate with different devices:

- System Driver: Supplies an additional port for the data exchange at the system level that normally happens on port 3246 (port 5000 for older systems).
- DataLink Driver: Supplies an additional port for the data exchange from DataLink that normally happens on port 3247 (port 5001 for older systems).
- Robotic Drivers for Motoman and ABB Robots.

The user can configure the Vision Sensor up as either a TCP/IP server or client and specify the preferred port. The user can also set the Ethernet port for communication as well as the type of communications driver to use on the port from this selection. There are two different methods of configuring an Ethernet Terminal Controller. The first one makes the Vision Sensor a TCP server, expecting a connection on port 5003. In this port it exposes the System Driver, which is the driver that receives most Ethernet commands to control the system. The second configuration is as a TCP client. In this case the system connects to a server with IP address 192.168.0.242 on port 5004 and sends DataLink strings to that

server. This is indicated by the selection of the DataLink Driver. This example demonstrates the two main types of configuration (server and client).

After this is properly setup the communications should be established when the Ethernet Terminal Controller is started or on power-up.

Industrial Protocols

DVT systems support other methods for communications such as standard protocols EtherNet/IP, ProfiBus and DeviceNet. EtherNet/IP support must be enabled from the I/O menu in Intellect and the power must be cycled for the changes to take effect. It is disabled by default to avoid loading the processor with operations not used. Using scripts, the user can interchange data with other devices that support the protocol. For more information about this type of connectivity please refer to the Script documentation and DVT integration notes. SmartLink, a different DVT product that can be connected to a DVT Vision Sensor can communicate via DeviceNet or PROFIBUS using expansion cards designed for that purpose. For more information about SmartLink, please refer to the SmartLink User Manual.

VDX Driver

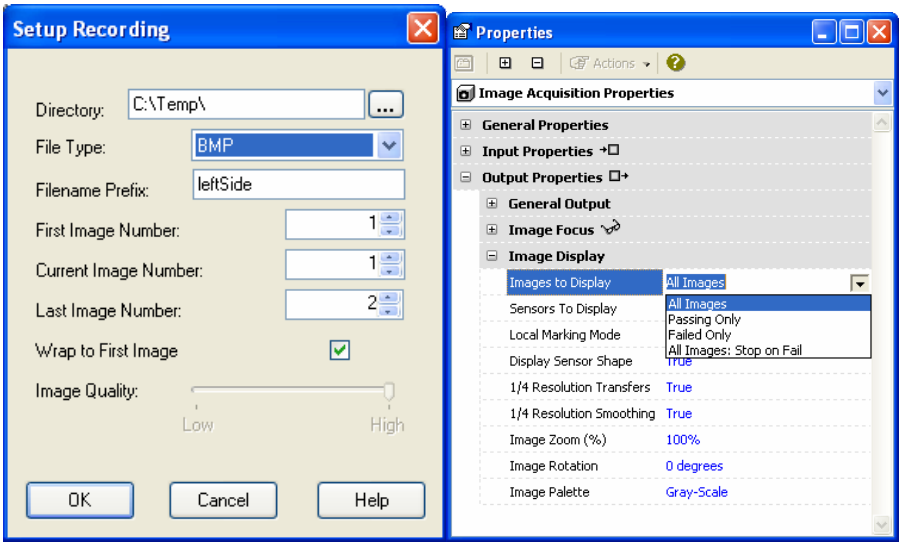
The VDX (Vision Data Exchange) driver is used as an alternative to Modbus to communicate between Vision Sensors and PROFIBUS or DeviceNet enabled SmartLink devices. The SmartLink device, as an intermediary, will exchange information with a PROFIBUS or DeviceNet master through these networks and will use the VDX driver to relay the information to the Vision Sensor. The VDX driver is enabled/disabled through the respective options under the I/O menu in Intellect. For more information about the protocol, consult the SmartLink User Manual, the Intellect Help Files and the Script Reference Guide.

Appendix A – Emulator tutorial

Chapter 2 – A Closer Look at the System explains the basics about the use of the Emulator. This appendix walks the user through a full example on how to work with it. Every step is explained and the Intellect dialog boxes are included for reference. If the user already has a set of prerecorded images and a product or system file the first 2 steps can be skipped.

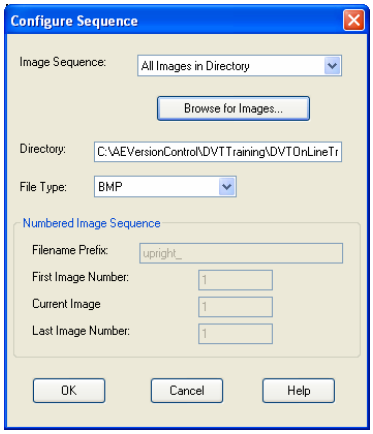
Step 1: Connect Intellect to the Vision Sensor and create a backup copy of the System or Product files. In order to do this, the user must select the desired option ("Save Product As" or "Save System As") from the "File" menu. This will prompt the user for a file name just like in any Windows® application. A file will be saved to the computer hard drive with the name specified by the user and an extension indicating if it is a system file (extension "dvtisys") or a product file (extension "dvtiprod").

Step 2: Record images from the Vision Sensor. In order to do this the user must select "Record->Recording Setup" under the "Images" menu. This dialog box lets the user select the directory where the images are to be saved, the name for the images and the number of images to record.

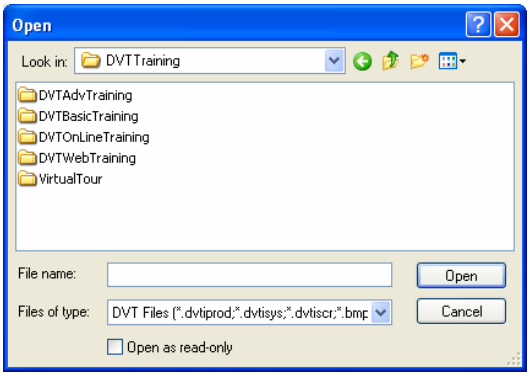


The user must specify the number of the first image in the sequence, the number of the current image (the next image to be recorded) and the last image to record. The last option "Wrap to First Image" indicates whether to stop on the last image or start overwriting from the first one. This will create a number of bitmap or jpeg images in the selected directory with the name and 3 digits indicating the image number. The setup shown in the dialog box above would save the following images: leftSide005.bmp, leftSide006.bmp, leftSide007.bmp, leftSide008.bmp, leftSide009.bmp, and leftSide010.bmp in the "C:\Temp" directory. This procedure saves every image that Intellect brings from the Vision Sensor. In the Image Acquisition properties window, the user can select which images to bring to the user interface. If only the failed images are to be saved, for example, the user should select to display Failed Images only. The Real Time Feedback should be started as usual from Intellect. When the user wants to start recording, the red circle "Record" button should be pressed from the image toolbar.

Step 3: Once the Product or System file and the images are available from the hard drive, the user can start Intellect and connect to an Emulator. The Emulator should be selected to match the Vision Sensor from which the images or Product/System files originated. Upon connection to the Emulator, Intellect will show the SID and make the standard editing options available. At this point the user can load the image sequence and the Product or System file. In order to load the image sequence, the user needs to select the option "Configure Sequence" under "Images". The following dialog box will appear:



Here the user has two options: type everything manually or click on the "Browse for Images…" button to find the desired images. In order to load the Product or System file, the Open action from the "File" menu should be selected.



With the images loaded, the user can make any change (that does not depend on physical setup) to the system just like if Intellect was connected to a Vision Sensor. When the user is done, step 1 should be repeated but in this case for the Emulator. A system or product file can be saved from the Emulator in the same way that it can be saved from the sensor. This procedure allows the user to set up the inspection parameters in the Emulator, and then load those as a System or Product file into the Vision Sensor.

Note: If users want to run the Emulator but they have no images, a standard graphics program could be used to create some images. The images need to be 256-color bitmaps of the same resolution as the Vision Sensor to be used (640 by 480 or 1280 by 1024). The naming convention should be followed as well.

Appendix B - Basic TCP/IP Setup

In this section, we will step through setting up the TCP/IP protocol on a PC. TCP/IP stands for Transmission Control Protocol / Internet Protocol and is based on identifying elements on a network by unique numbers, called IP numbers.

We will assign our PC with one IP number and any Vision Sensor with another. There are several methods of connecting PCs and Vision Sensor systems. Note that if the PC is already in a network, it perhaps is configured properly and does not require a TCP/IP setup.

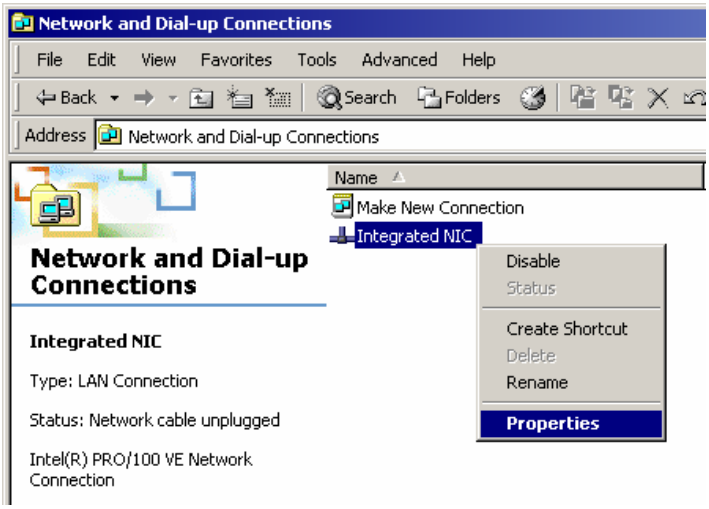
For Windows XP and Windows 2000®:



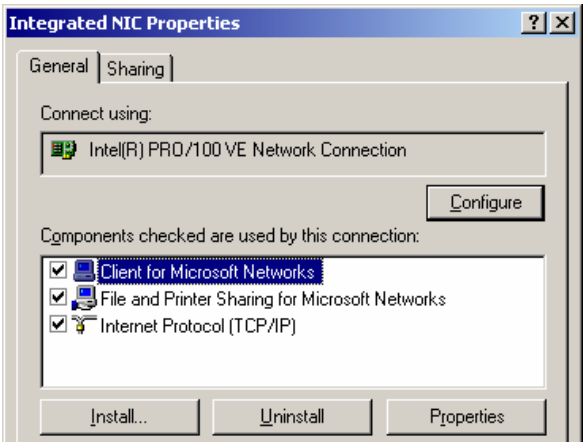
Click on the **Start** menu in Windows®, select **Settings**, then **Control Panel**.

Double click on the **Network and Dial-Up Connections** Control Panel icon (shown above right).

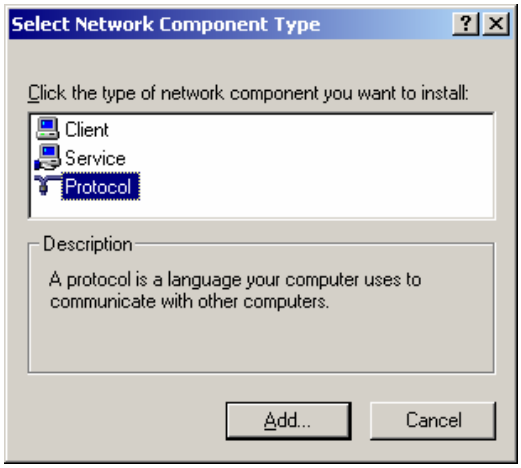
To check for the list of installed network components for your network card, right-click on it and select **Properties** (as shown below). If TCP/IP is on the list in the corresponding dialog box, click **Cancel** and skip to the Communicating with the Vision Sensor section.



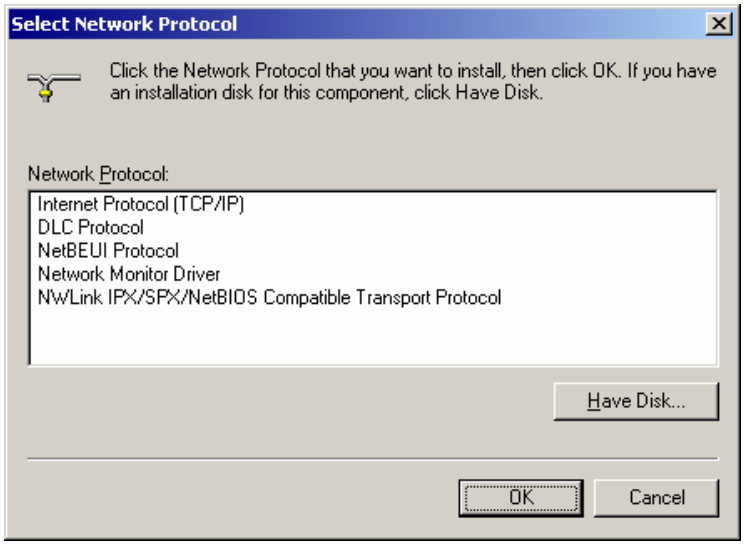
If TCP/IP is not on the list, click the **Install...** button.



Click on **Protocol**, then **Add...** (shown below).



Select **Internet Protocol (TCP/IP)** then click **Ok** (see below).



Windows® will then install the protocol and return you to the Network control panel. During this installation process, the PC may ask for the Windows 2000® CD-ROM and will copy relevant files to the PC's hard drive. After this is done, restart your PC and continue to the Communicating with the Vision Sensor section.

Appendix C - Advanced TCP/IP

The Vision Sensor runs TCP/IP over an Ethernet (RJ-45 only) connection. This Appendix provides some advanced information about this setup that may be useful to Network Administrators.

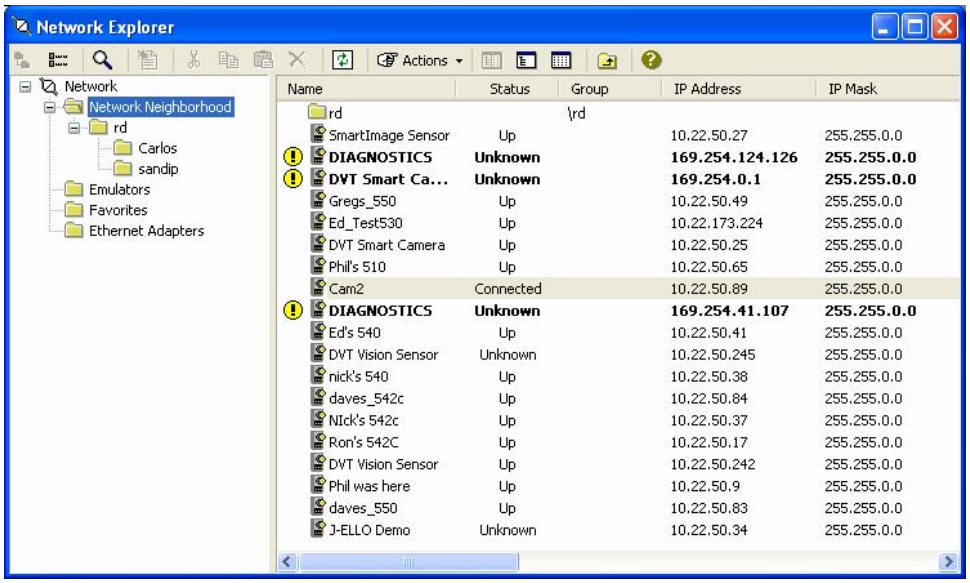
TCP/IP Installation Tips

DVT Vision Sensors are shipped with a random IP address, which must be changed according to your network. When assigning an IP address to the Vision Sensor, you need to make sure that it is one that has not already been assigned to another device on the network.

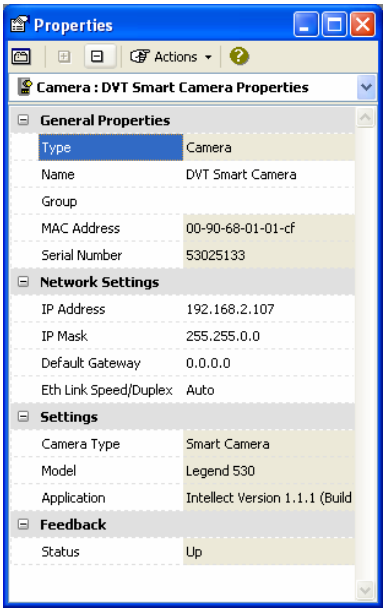
Vision Sensors do not require a Default Gateway to be set. For more information on what a Default Gateway is, please refer to the respective section later in this appendix.

Typically, when connecting your PC to the Vision Sensor, the network masks must match.

The IP address can be changed from Intellect software (**System** menu, **Network Explorer** selection), or from the Network Neighborhood Browse (see below).



Highlighting the appropriate DVT Smart Sensor in the right pane causes the **Vision Sensor's Property** window to appear. The Vision Sensor's Property window allows the Vision Sensor's IP Address and subnet mask to be changed.



TCP / IP Troubleshooting Tips

IP addresses for each PC (and for each network card installed on the PC) and each device that uses the TCP/IP protocol (such as the Vision Sensor) is required to have a unique IP address.

IP addresses should not use 0 or 255 as the last value (e.g. 192.168.1.0 and 192.168.1.255, respectively). These are reserved and have special meaning:

Addresses that end in 0 specify all addresses for a particular subnet.

Addresses that end in 255 specify a broadcast packet that is sent to all hosts on the network.

With both PC and Vision Sensor using the Subnet (or Address) mask of 255.255.255.0, the IP addresses of the PC and the DVT Vision Sensor must be identical for the first three values and unique in the fourth value. **Example**: DVT Vision Sensor (IP: 192.168.0.242) & PC (IP: 192.168.0.240). Please refer to the section on subnetting later in this appendix.

To find the DVT Vision Sensor's current IP Address, note the Vision Sensor's serial number on the back of the sensor's case and find the appropriate DVT Vision Sensor serial number in the serial number column of the Network Explorer. Note the corresponding IP address in the IP address column.

Type 'ping ###.###.###.###' (replacing # with numbers) to check if an IP address is connected to the network. This command can be useful in determining if a node is powered up, operational, and physically connected to the network.

Type 'arp –a' from a DOS prompt to view the ARP (Address Resolution Protocol) table. The response should look something like this:

```
C:\WINDOWS>arp -a

Interface: 192.0.0.111
  Internet Address      Physical Address      Type
  192.0.0.211           00-90-68-00-00-42     dynamic
  192.0.0.212           00-90-68-00-00-42     dynamic
  192.0.0.253           00-60-97-4f-c0-45     dynamic
```

The "Internet Address" column refers to the IP addresses you've recently communicated with. The "Physical Address" column contains the corresponding MAC Addresses for the IP's in column 1. DVT Vision Sensors will have Physical Addresses starting with "00-90-68". In the case above, 192.0.0.253 is not a Vision Sensor, but another PC on the same network as the named Interface. The ARP table is cleared approximately every two minutes. The 'arp' command is useful in determining if an IP address belongs to a Vision Sensor system or another node on the network.

TCP/IP Subnetting

Please note that the creation of custom subnets is beyond the scope of this appendix and so will not be discussed. This section will only discuss default subnets (i.e. 255.0.0.0, 255.255.0.0, and 255.255.255.0), some key terms relating to subnetting, and some basic rules on setting up IP addresses with the default subnet masks.

Subnetting is a common technique among network administrators to segregate a network so that network traffic remains in certain areas (or subnets). If a device in one subnet needs to communicate to another device in another subnet and both devices reside on the same physical network, a router will be required (see the definition of a Default Gateway) to provide that link between the two areas. However if the two devices reside in the same subnet, then no router is required and the two devices can communicate with each other. As a rule of thumb, if the subnet mask is 255.255.255.0 on all devices, then the first three numbers of an IP address, which is the Network ID (e.g. **192.168.203**.1), will need to be the same and the last number, which is the Host ID (e.g. 192.168.203.**1**, 192.168.203.**2**, 192.168.203.**3**, etc.), will need to be unique in order for those devices to communicate with each other. Similarly, if the subnet mask is 255.255.0.0 on all devices, then the first two numbers of an IP address will need to be the same and the last two can be different.

Below are some key terms related to subnetting:

Physical Segment – A Physical Segment (or subnet) is a portion of the network that can receive a broadcast packet. All computers within a specified Physical Segment will share a common Network ID. Physical Segments are separated by routers because routers cannot forward broadcast packets; they can, however, traverse hubs, switches, and bridges.

Network ID – The Network ID is the part of an IP address that is common to all computers on a specified physical segment. The Network ID is equivalent to the area code of a phone number.

Host ID – The Host ID is the part of an IP address that uniquely identifies the device on a specified physical segment. The Host ID is equivalent to the phone number.

Default Gateway – A Default Gateway is typically a router that provides a route between two Physical Segments. It does not forward broadcast packets. So, if your Vision Sensor resides in one Physical Segment and your PC resides in another Physical Segment, then you will need to setup a Default Gateway to provide a route between the two Physical Segments. You would then need to setup up your Vision Sensor and your PC to point to that Default Gateway. Specifying a Default Gateway is not required, especially if both your Vision Sensor and your PC reside on the same subnet.

Subnet Mask (SNM) – A Subnet Mask is the second element required in TCP/IP communications, along with the IP address. Without a subnet mask, communication will not be able to take place between two network devices. The Subnet Mask also helps you to identify what is the Network ID and what is the Host ID in an IP address.

Appendix D - Upgrading or Reinstalling DVT Vision Sensor Firmware

Before installing Intellect on your PC and upgrading the firmware file in the DVT Vision Sensor, take the time to backup the existing system to the PC.

To back up a DVT Vision Sensor system to a PC:

- Run the older version of Intellect and connect to the DVT Vision Sensor.
- Click on the **File** menu.
- Choose Save System As in the File menu
- When prompted, provide a filename.
- Exit Intellect.

Now that the old system has been fully backed up, Intellect may be installed on the PC.

To install Intellect on a PC:

- Insert the Intellect CD. The installation should automatically start.
- If the CD does not automatically start, do the following:
    - From the **Start** menu, choose **Run**.
    - Type **D:\Setup.exe** (substitute the appropriate drive letter of your CD-ROM drive for **D**) and press **Enter**.

OR

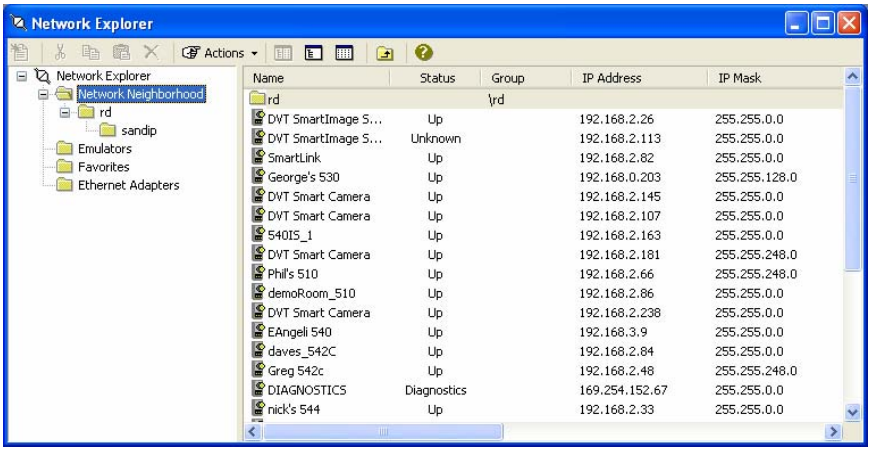
- Run the installation file downloaded from the DVT website
- Follow the installation instructions on screen.
- After Intellect has been installed on the PC, the firmware file on the DVT Vision Sensor must be upgraded.

Upgrading Intellect firmware on a DVT Vision Sensor

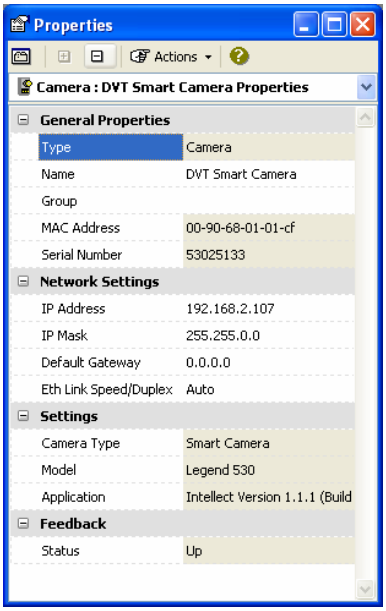
Note: The DVT Vision Sensor is a stand-alone unit that runs the firmware. All DVT Vision Sensors arrive from the factory with Intellect firmware installed. This step should not be necessary unless reinstalling or upgrading firmware.

Before upgrading, start Intellect and click Help|About to verify the version of the software being used.

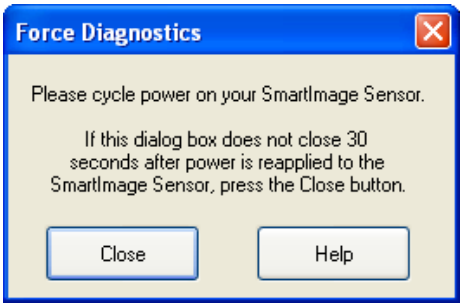
1. Start Intellect. The "Network Explorer" window come up.
2. Select "Network Neighborhood". This will search your local network for DVT Vision Sensors.



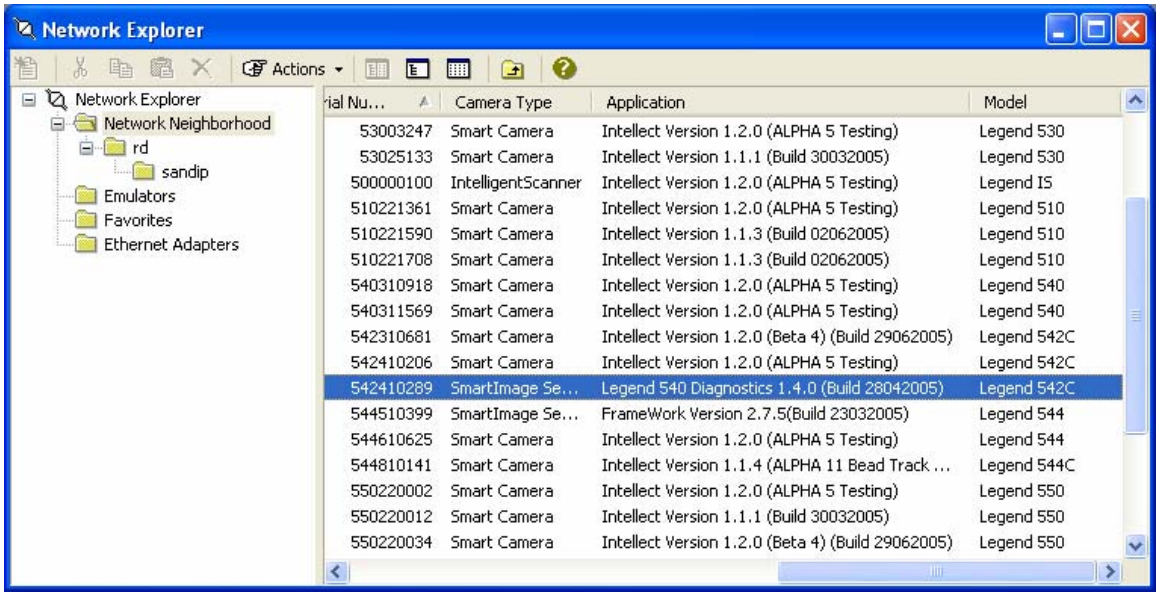
3. Highlight the appropriate sensor by choosing the system with the matching serial number and make sure the IP address and subnet mask are compatible with your computer's IP address and subnet mask.
4. If you need to change the IP address and/or the subnet mask of the DVT Vision Sensor, select the appropriate file for the Vision Sensor Properties window and change the IP address and subnet values.



5. It is recommended that before loading firmware into the system, the user erases the content of flash memory. This will avoid potential problems that could arise when loading different firmware versions on top of one another. The first step to do this is forcing the system into diagnostics mode. Select the appropriate DVT Vision Sensor under "Network Neighborhood" by right mouse button clicking and selecting "Force Diagnostics". Intellect will instruct you to cycle power on the system you are trying to upgrade.

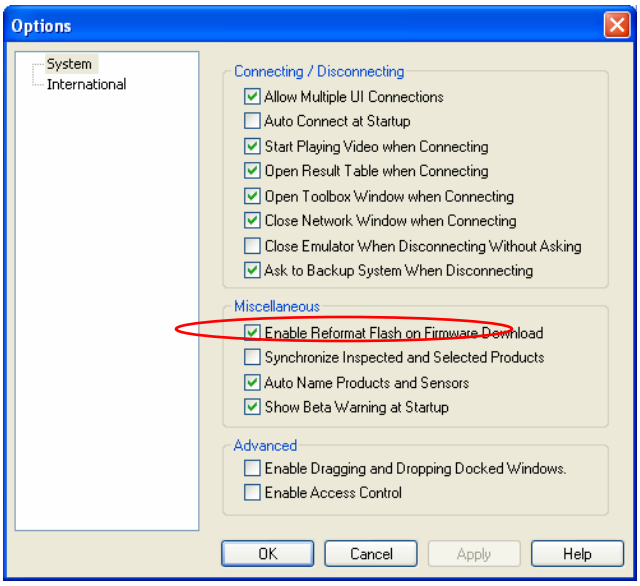


6. After the system boots up it should be in diagnostics mode. You can check this by verifying that the "Status" LED on the back of the Vision Sensor is blinking red. Also under "Network Neighborhood", the system should appear now as "DIAGNOSTICS".
7. When the system is forced into diagnostics mode, it resets its IP address. This means that at this point you would need to select the system, click on the "Edit…" button and reassign the appropriate IP address and net mask.
8. Now that the system is in diagnostics mode you can proceed to erase flash memory. While the Vision Sensor is in diagnostics, look in the Application Column of the "Network Explorer" table and identify the diagnostics version of the system.



9. If this version is 0.28 or a more recent release (like in the case above) you would be able to erase flash memory and load the new firmware version from Intellect. If your system has an older diagnostics version go to the section "Erasing flash using Telnet" at the end of this appendix and then continue on step 11. The mentioned section explains a fail-safe method to erase flash memory; so if your system has a recent diagnostics version but the procedure described below to accomplish this task does not work, you can still refer to the technique covered at the end of the appendix to achieve your objective of clearing the contents of permanent memory.

10. Before proceeding, make sure that the option "Enable Reformat Flash on Firmware Download" is enabled in Intellect. You could verify this by closing the "PC Communications" dialog box and clicking on "Options…" under the "Edit" menu in Intellect. The option is indicated in the next figure.



11. On the "Network Explorer" window, select the DVT Vision Sensor on the list and right mouse button click and select the "Load Firmware" menu. Select the firmware version that you want to load based on your model. The firmware files for all the systems are included in the Intellect installation and their extensions identify the sensors for which they were designed. At this point, if you are erasing flash from Intellect, you should select "Yes" when the corresponding message appears.
12. Wait for approximately 30 seconds. If the progress bar closes before that, make sure the correct Vision Sensor is highlighted in the right hand pane and right mouse button click on the DVT Vision Sensor and select the "Connect" option.
13. Before getting into Intellect, the program will ask you if you would like to store the firmware to Flash Memory. Choose Yes. It will take a few seconds to store the file to flash memory. Then the Sampled Image Display will appear in the User Interface of Intellect.

Erasing flash using Telnet

This is a fail-safe method to erase flash memory. You should read this section and follow the instructions explained if the diagnostics version of the system you are trying to upgrade is older than version 0.28 or if the procedure covered in steps 10-11 above does not work.

After forcing the system into diagnostics mode and reassigning the appropriate IP address and net mask, you can proceed to clear the contents of flash memory. You should use telnet or any other terminal emulator to connect to the DVT Vision Sensor through Ethernet on port 23 (which is the default port for telnet.) Telnet is included as an accessory in all versions of Windows®.

- Click on "Start" then "Run…"

- Type "telnet" and press Enter.

    - In Windows 2000®, XP and NT telnet is included as a DOS-based program. In this case you will see a DOS console appear. At the prompt, type "open" followed by a space and then the Vision Sensor's IP address.

- After you are connected, you will see some information about the system displayed. At the question mark prompt, type the following command without the quotes: "*F1bdc" (You might not see what you are typing if you do not have local echo enabled in the telnet program.)

- You should see the message "Erasing Flash". After the process is completed, you will receive the question mark prompt. At this time, permanent memory has been erased and you should close the telnet program.

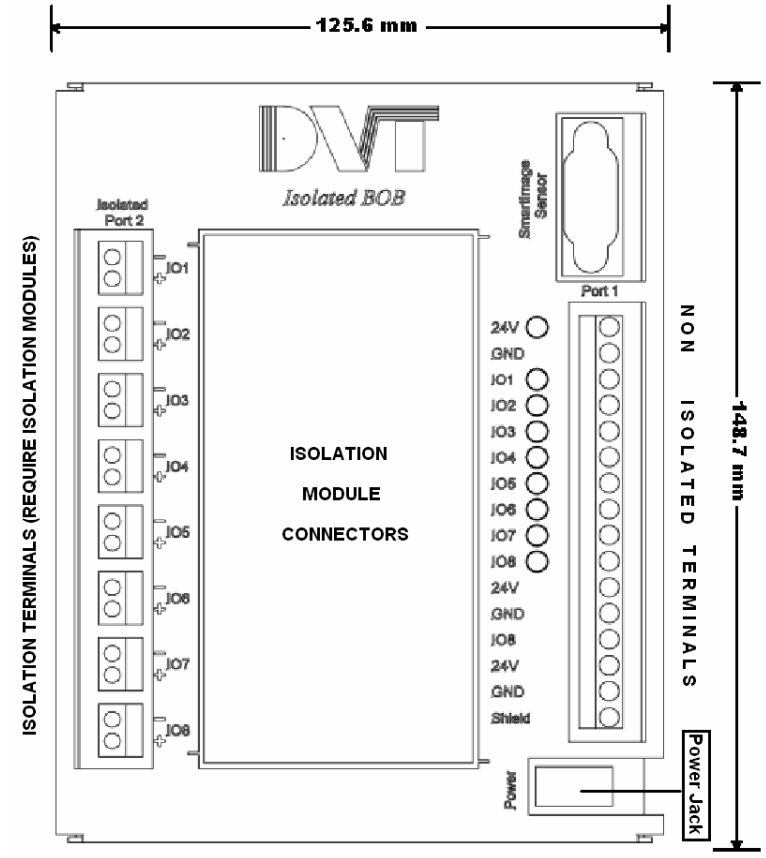
Appendix E – Breakout Boards

Isolation Breakout Board

The isolated breakout board for the DVT Vision Sensors not only provides a convenient method to connect digital I/O, power, and strobe illumination lines; but it also enables the use of isolation modules. The board is housed in a metal/plastic DIN Rail-mountable case with powder coating on top. The breakout board is 126mm wide by 149mm high by 45mm deep (including all necessary cable clearances.) The board has two main ports. The port on the right includes I/O, strobe and power screw-down terminals. The I/O lines on this port are to be used when no isolation is necessary. The second port, on the left side, has the I/O connections to be used in conjunction with isolation modules. The optional isolation modules can be acquired from third-party companies to provide power isolation when working with devices that are using a separate power supply.

Features	Purpose
Power Jack	Included for the use of a 24V (1.42 A) wall-mount power supply. It should only be used for demonstration and laboratory purposes.
Power Terminals	There are three main power terminals (V+, GND, Shield) providing connections for an industrial power supply. There are two additional pairs of V+ and GND terminals.
Non-isolated I/O Terminals	Located on the right side of the board there are screw-down terminals for the 8 I/O lines available on the DVT Vision Sensors. There is an extra terminal connected to I/O line 8 intended to provide one more output for a strobe illumination. This output will be activated during image exposure only if it is configured in Intellect as the Strobe output.
Isolated I/O Terminals	Located on the left side of the board, these are to be used in combination with isolation modules.
Isolation modules slots	There are 8 screw-down slots for input or output isolation modules.
15 pin high density D-sub	This provides connection to the I/O and power lines of the DVT

plug	DVT vision systems which use a 10-pin keyed RJ-45 port.



Index

1D Barcode Reader **76**
2D Reader **78**
Access Control
Password Protection 29
Antiblooming **36**
Background Scripts **32, 49**
Blob Counting **60**
Blob Locating **67**
Close **42**
Color **83**
Color Monitoring **86**
Communications
DataLink 102
Ethernet Terminal Controller 104
Industrial Protocols 105
Modbus Master 103
Modbus Transfers 103
Dark Image Subtraction **33**
DataLink **102**
Datalink Window **49**
Digital I/O Configuration **30**
Digitizing Time **35**
Dilate **42**
Drivers **104**
DVT Spectral Sensor **34**
DVT Vision Sensor **10**
Dimensions 15
Hardware Setup 21
Setup 21
System Components 27
DVT Vision Sensors **10**
Edge Width **70**

EdgeCount **57**
Emulator **52**
Erode **42**
Ethernet Communications
Drivers 104
Ethernet Terminal Controller **104**
Exposure Time **34**
FeatureCount **58**
Foreground Scripts **87**
FOV Balance **33**
Getting Started **9**
Hardware Setup **21**
Identification Vision Tools **76**
Illumination **36**
Industrial Protocols **105**
Inspection mode **29**
Inspection Mode **50**
Integration **101**
Intellect Software **48**
Intellect User Interface **48**
Main Menu 48
Result Table 51
Status Bar 51
Toolbar 50
Video Display 51
Vision Tool Toolbox 51
Intensity Vision Tool **59**
IP Address **17**
Line Fit Vision Tool **66**
Machine Vision **10**
Imaging 10
Pixels 10
Math Operations **74**
Max Iterations **71**
Measurement **44, 70**
Measurement in Area **70**

Measurement in Circle **70**
Minimum Contrast **44**
Modbus Master **103**
Modbus Transfer **103**
Object Counting **60, 63**
Object Locating **67**
OCR **79**
Open **42**
Outlier Distance **72**
Partial Image Window **35**
Password Protection **29**
Pattern Counting **60, 64**
Pattern Locating **67**
PatternCount **59**
Pixel Counting **83**
Pixel Counting Vision Tool **59**
Pixel Graph **45**
Pixels **10**
Play mode **29**
Points and Lines **74**
Power-on Product **32**
Preprocessing **69**
Product parameters
Product Graphs 38, 40
Product Parameters **34**
Antiblooming 36
Digitizing Time 35
Exposure Time 34
Illumination 36
Partial Image Window 35
Product ID 37
Sensor Gain 33
Product Selection **37**
Readers **76**
1D Barcode Reader 76
2D Reader 78

OCR 79
Rotation 66
Script Debug Window 49
Sensor Gain 33
Sensor Graphs 45
Spectrograph 88
Spectrum Calibration 34
Statistics 74
System Parameters 29
Background Scripts 32, 49
Dark Image Subtraction 33
Digital I/O configuation 30
FOV Balance 33
Input Functions 30
Inspection mode 29
Inspection Mode 50
Output Functions 31
Power-on Product 32
Reflectance Calibration 34
Spectrum Calibration 34
Trigger 33
White Image Normalization 33
System Specifications 12
Template Matching 64
Terminal Window 49
Threshold 43
Adaptive 44
Auto Bimodal 44
Dominant Color 83
Fixed Value 43
Gradient 44
OCR 80
Percent of Path Contrast 43
Percent of Reference Intensities 44
Training 24
Translation 65

Trigger Source **33**
Use Best Edge **70**
Vision Sensor
Product Parameters 34
System Parameters 29
Vision Tool Parameters
Digital Relearn 47
Image Processing 42
Result 44
Sensor Graphs 45
Shape 41
Threshold 43
Vision Tools **55**
Area Positioning 67
Color 83
Color Monitoring 86
Color Space 83
Count in Area 60, 63, 64
EdgeCount 57
FeatureCount 58
Identification 76
Intensity 59
Line Fit 66
Math Tools 74
Measurement 44, 70
PatternCount 59
Pixel Counting 83
Preprocessing 69
Readers 76
1D Barcode Reader 76
2D Reader 78
OCR 79
Scripts (Foreground) 87
Spectrograph 88
Translation 65
Vision Tools Parameters **41**

White Image Normalization 33

Artisan Technology Group is an independent supplier of quality pre-owned equipment

Gold-standard solutions

Extend the life of your critical industrial, commercial, and military systems with our superior service and support.

We buy equipment

Planning to upgrade your current equipment? Have surplus equipment taking up shelf space? We'll give it a new home.

Learn more!

Visit us at **artisantg.com** for more info on price quotes, drivers, technical specifications, manuals, and documentation.

Artisan Scientific Corporation dba Artisan Technology Group is not an affiliate, representative, or authorized distributor for any manufacturer listed herein.

We're here to make your life easier. How can we help you today?

(217) 352-9330 | sales@artisantg.com | artisantg.com

ARTISAN® TECHNOLOGY GROUP