ABB

ROBOTICS

Application manual

Integrated Vision

Trace back information:
Workspace Main version a654
Checked in 2025-03-10
Skribenta version 5.6.018

Application manual

Integrated Vision

RobotWare 6.16

Document ID: 3HAC044251-001

Revision: R

Keep for future reference.

Additional copies of this manual may be obtained from ABB.

Original instructions.

Table of contents

Overview of this manual

About this manual

This manual contains instructions for installation, configuration, and daily operation of the option Integrated Vision.

Usage

This manual should be used during installation, configuration, and maintenance of a system with the option Integrated Vision.

Who should read this manual?

This manual is intended for:

- Installation personnel
- Programmers
- Operators

Prerequisites

Maintenance/repair/installation personnel working with an ABB robot must be trained by ABB and have the required knowledge of mechanical and electrical installation/repair/maintenance work.

References

Reference	Document ID
Operating manual - RobotStudio	*3HAC032104-001*
Operating manual - IRC5 with FlexPendant	*3HAC050941-001*
Technical reference manual - RAPID Instructions, Functions and Data types	*3HAC050917-001*
Technical reference manual - RAPID Overview	*3HAC050947-001*
Technical reference manual - System parameters	*3HAC050948-001*
Product manual - IRC5 *Product manual - IRC5 Compact* *Product manual - IRC5 Panel Mounted Controller* IRC5 with main computer DSQC 639.	*3HAC021313-001* *3HAC035738-001* *3HAC027707-001*
Product manual - IRC5 *Product manual - IRC5 Panel Mounted Controller* *Product manual - IRC5 Panel Mounted Controller* IRC5 with main computer DSQC10xx.	*3HAC047136-001* *3HAC047138-001* *3HAC047137-001*
Product manual - Product.ProductName	*Document.ID-1*
Cognex In-Sight® 7000 Series Vision System Installation Manual This manual describes how to install the vision camera. Applicable for cameras DSQC1020, DSQC1021, and In-Sight Micro 1402.	*P/N 597-0138-01*
In-Sight® 7000 Gen2 Series Vision System, Reference Guide This manual describes how to install the vision camera. Applicable for cameras DSQC1063, DSQC1064, DSQC1098 and DSQC1099.	-

Continues on next page

Revisions

Revision	Description
-	Released with RobotWare 5.15.01. First release.
A	Released with RobotWare 5.15.03 and RobotWare 5.60. Updates and corrections throughout the manual, among others: • The list of limitations is updated, see *Introduction to Integrated Vision on page 11*. • The installation chapter is updated and restructured, see *Installation on page 15*. • The options dialog is added, see *Options dialog on page 30*. • The language settings for vision parameters is changed, see *Changing the language on page 32*. • Information about removing a camera, updating the camera firmware, and connecting to a camera emulator is updated and added, see *Setting up the camera on page 41*. • The best practise chapter is updated with multiple new sections, see *Best practise on page 79*.
B	Released with RobotWare 5.61. Updates and corrections throughout the manual, among others: • The error recovery `ERR_CAM_COM_TIMEOUT` is added to several RAPID instructions.
C	Released with RobotWare 6.0. • New functionality to restrict user access, see *Restricting user access on page 47*. • Preprinted calibration plates are now available, see *Camera calibration on page 54*. • The best practise chapter is updated with a new section, see *Always check that the vision target is within expected limits on page 95*. • Minor corrections.
D	Released with RobotWare 6.01. • Updated references to other manuals, see *References on page 7*. • Updated the list of required hardware and software, see *Checklist on page 13*. • Updated the installation procedure, see *Installing the hardware on page 15*. • Minor corrections.
E	Released with RobotWare 6.02. • Updated the information about *Image trigger on page 52*. • Added the argument `\AwaitComplete` to the RAPID instruction `CamReqImage`. • The RAPID instructions, functions, and data types are now also available in *Technical reference manual - RAPID Instructions, Functions and Data types*.
F	Released with RobotWare 6.03. • Removed the RAPID instructions, functions, and data types from this manual. They are now only described in *Technical reference manual - RAPID Instructions, Functions and Data types*. • Minor corrections.
G	Released with RobotWare 6.04. • Minor corrections.
H	Released with RobotWare 6.05. • Minor corrections.

Continues on next page

Revision	Description
J	Released with RobotWare 6.06. • Minor corrections in *The RobotStudio user interface on page 17*. • Minor corrections in *Configuring Integrated Vision on page 39*.
K	Released with RobotWare 6.08. • Minor corrections in *RAPID components on page 97*.
L	Released with RobotWare 6.12. • Added new cameras, Cognex In-Sight® 7600/7800 • Added the new instructions `CamGetMode`, `CamStartSetParameter`, and `CamWaitSetParameter`. • Added the data type `camerastatus`. • Added limitation for the instruction `CamSetParameters`.
M	Released with RobotWare 6.13.02. Camera emulator removed.
N	Released with RobotWare 6.14. Updated Camera firmware version.
P	Released with RobotWare 6.15.03. • Added information about FlexPendant authorization in *Restricting user access on page 47*. • Removed limitation regarding 64-bit version of RobotStudio. • Addition of new cameras Cognex In-Sight 7900.
Q	Released with RobotWare 6.15.08. • Updated the section *Using the `SceneId` argument on page 69*.
R	Released with RobotWare 6.16. • Added section *Firewall Settings*. • Minor updates

1 Introduction to Integrated Vision

1.1 System overview

What is Integrated Vision?

The purpose of ABB's Integrated Vision system is to provide a robust and easy-to-use vision system for general purpose Vision Guided Robotics (VGR) applications.

The system includes a complete software and hardware solution that is fully integrated with the robot controller and the RobotStudio programming environment. The vision capability is based on the *Cognex In-Sight®* smart camera family, with embedded image processing and an Ethernet communication interface.

The Integrated Vision add-in in RobotStudio is equipped with a vision programming environment that exposes the palette of *Cognex EasyBuilder®* functionality with robust tools for part location, part inspection, and identification.

Hardware

The camera system is based on the *Cognex In-Sight® 7000 series*, but most *Cognex In-Sight®* cameras can be used. The camera is supplied with 24 VDC and Ethernet from the controller.

The cameras are connected to the supplied Ethernet switch. The maximum number of cameras are three.

For more information see *Cognex In-Sight® 7000 Series Vision System Installation Manual* and *In-Sight® 7000 Gen2 Series Vision System, Reference Guide*.

Note

The functionality in Integrated Vision is verified with the following cameras (with ABB numbers):

- DSQC1020, DSQC1021, and In-Sight Micro 1402 (ABB branded)
- DSQC1063 and DSQC1064
- DSQC1098 and DSQC1099

For other camera models all features might not work, for example, color is not supported.

Software

RobotStudio with the Integrated Vision add-in presents vision and robot configuration parameters side by side, providing a convenient VGR programming environment.

1.2 Vision safety

General principles

Using a vision sensor for robot guidance requires that the user observes caution when handling, installing, and configuring the system.

The user must always assume that the vision sensor is active even if the manipulator is not moving.

Before entering the working range of the manipulator, the user must take the following precautions to prevent the manipulator from starting to move.

WARNING

If work must be carried out within the manipulator's work area, the following points must be observed:

- The operating mode selector on the controller must be in the manual mode position to render the three-position enabling device operational and to block operation from a vision sensor, a computer link, or a remote control panel.
- Anyone entering the manipulator working space must always bring the FlexPendant with him/her. This is to prevent anyone else from taking control of the manipulator without his/her knowledge.
- The three-position enabling device must never be rendered inoperational in any way.
- During programming and testing, the three-position enabling device must be released as soon as there is no need for the manipulator to move.

1.3 Getting started with Integrated Vision

Checklist

Before setting up the system, make sure that the necessary preparations have been made.

Hardware

- The option *Integrated Vision interface* which includes:
    - Ethernet switch.
    - Ethernet cable for connecting the camera to the switch.
    - Customer power supply.
    - Wires for supplying 24 VDC from the customer power supply to the Ethernet switch.
    - Cable for supplying 24 VDC to the camera.
- A pointing tool to be mounted on the robot for accurately defining work objects.
- PC.
- Ethernet cable for connecting the PC to the controller.
- *Cognex In-Sight®* camera.
- Camera lens.
- Lighting device.

Software

- RobotStudio 2025.1 or later, full installation.
- Integrated Vision add-in, version 1.0 or later.
- A RobotWare license with the option *Integrated Vision* enabled.

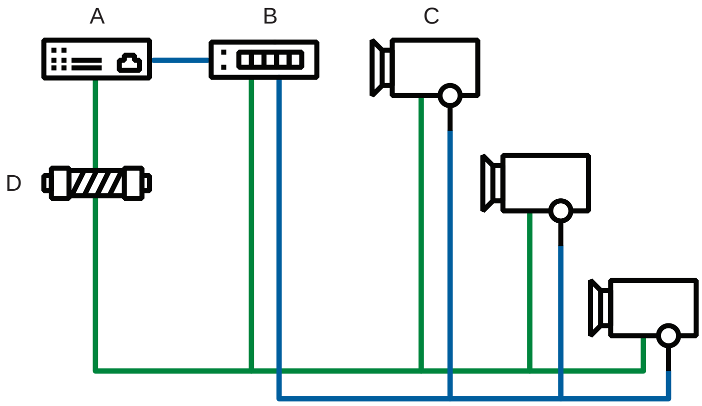
For information on how to configure Integrated Vision, see *Configuring Integrated Vision on page 39*.

This page is intentionally left blank

2 Installation

2.1 Installing the hardware

Overview



xx1900000617

A	Robot controller
B	Ethernet switch
C	Cameras
D	24V power supply

Installation procedure

Connect the following components and cables according to the figure:

For more information see the *Cognex In-Sight® 7000 Series* manual, the *In-Sight® 7000 Gen2 Series Vision System, Reference Guide* and the product manual and circuit diagram for the corresponding controller. See *References on page 7*.

CAUTION

When using a robot held camera, or by other means moving camera, it is important to have a good cable routing along the robot arm.

When routing the cables caution has to be taken to avoid mechanical stress on the connectors, allowing sufficient bend radius for the cables, and minimizing the wear on the cables. It is also recommended to fit the cables with extra wear protection at the attachment points and at especially exposed areas.

2.2 Installing the software

Installing RobotStudio

The Integrated Vision configuration environment is designed as a RobotStudio add-in, and is installed from the RobotStudio Gallery.

	Action
1	Install RobotStudio. Select complete installation.
2	Start RobotStudio.
3	Install the Integrated Vision add-in from the RobotStudio Gallery.
4	Go to the **Controller** tab on the ribbon menu and start the Integrated Vision add-in. After the add-in has finished loading a new tab named **Vision** is visible. **Tip** When a controller is connected, the Integrated Vision add-in can be started from the context menu of the controller node, the vision system node, or the camera node in the controller browser.

For more information see *Operating manual - RobotStudio*.

Note

It is not recommended to change the RobotStudio language after the configuration of an Integrated Vision system has started.

For more information, see *Changing the language on page 32*.

Tip

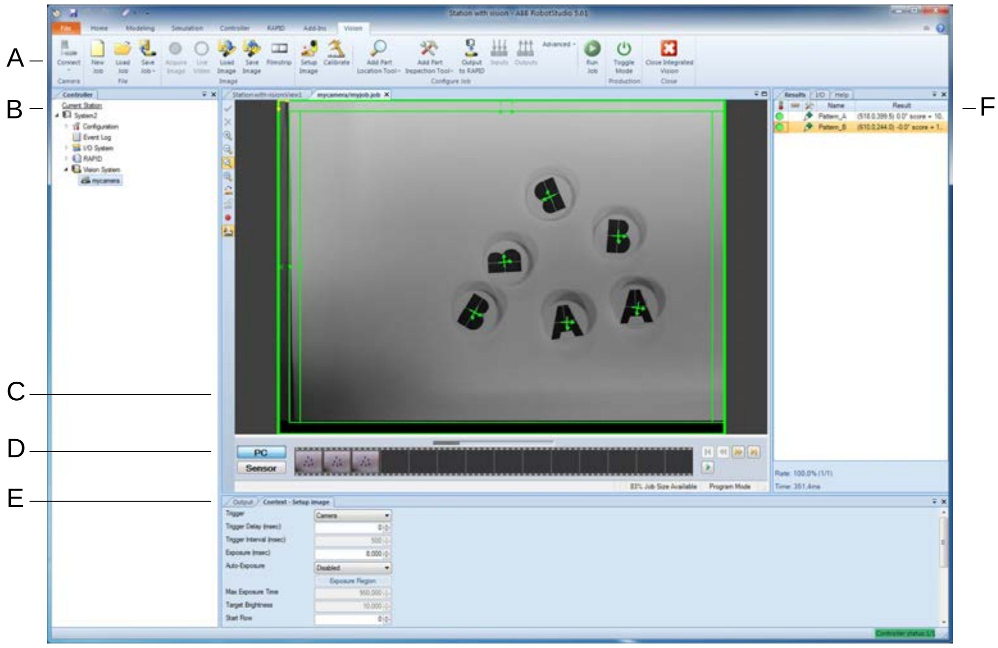
A 3D model of the vision camera is included in the RobotStudio library.

3 The RobotStudio user interface

3.1 The main window

Overview of the main window

This section presents an overview of the Integrated Vision graphical user interface.



xx1200000989

	Parts	Description
A	Ribbon	Displays groups of icons organized in a logical sequence of function.
B	Configuration browser	The Configuration browser displays the I/O configuration tree and corresponds to the I/O System configuration tree.
C	Image capture and configuration area	Displays an image acquired by the vision camera with configuration guides for locating and inspecting parts.
D	Filmstrip bar	Used to record a sequence of images for later analysis.
E	Context window	Contains the available properties, settings, and events of the selected controls.
F	Palette window	Following tabs are available: • **Results** tab - displays the setup of the active vision job with a list of all used location and inspection tools. • **I/O** tab - displays the I/O setup. • **Help** tab - provides online help.

Tip

If any window is accidentally closed, it can be restored from the **Customize Quick Access Toolbar**.

3.2 Online help

General

The experienced vision camera user, familiar with the *Cognex* line of products, will recognize most of the graphical user interface for vision in RobotStudio.

This effects the online **Help** tab, since the online **Help** is also integrated without modifications of the content.

All references to parameters, settings, and technical explanations are correct. But any references to the GUI can be slightly incorrect depending on the implementation in RobotStudio.

Therefore use the online **Help** tab as the technical reference manual for all parameters and settings, and use this application manual as the reference for the graphical user interface.

The online help tab

The online **Help** tab is located in the palette window.

The online **Help** tab is a context sensitive technical reference manual for all Integrated Vision parameters and settings. This means that the content of the help tab adjusts to the operation that is currently performed in Integrated Vision. For example setting up the image, calibrating, adding location tools, etc.

Note

Most parameters and settings are only described in the online **Help** tab, and not in this application manual.

Terminology

The list below describes the most common differences in terminology between the online help tab and the Integrated Vision application.

If the online help tab says...	...then it means:
EasyBuilder®	RobotStudio vision add-in tab
edit acquisition settings (group box)	the **Setup Image** button
calibrating the image to real world units (group box)	the **Calibrate** button
online	**Run Mode**
offline	**Program Mode**

3.3 The ribbon

Layout of the ribbon

The Integrated Vision tab contains groups of commands organized in a logical sequence of functions that simplifies the managing of Integrated Vision projects.

The tab consists of the following groups:

Group	Functions used for
Camera	Connecting to the vision cameras and setting camera properties.
File	Creating, saving and loading vision jobs.
Image	Loading and saving images. Images can both be acquired from a vision camera or loaded from a previously saved file.
Configure Job	Work flow of configuring vision jobs, organized in a logical sequence of functions.
Production	Setting the Integrated Vision system in program mode or run mode. (Similar to the robot controller auto/manual key switch on the control panel.)

Camera group

The **Camera** group is used when connecting to cameras and configuring cameras.

Button	Description
Connect	Connect to the selected camera.
Disconnect	Disconnect the selected camera. **Note** A camera is also disconnected when the corresponding tab in the image capture and configuration area is closed.
Rename	Rename the camera to make it available in RAPID with a name.
Network Settings	Change the IP-settings of a connected camera.
Set Date and Time	Set the date and time of the camera.
User Access Settings	Edit the list of users on the camera and their associated privileges.
Set Controller User	Select the user profile to be used by the controller when communicating with the camera.
Add Sensor	Find a camera on the network and change its IP-settings.
Properties	View the camera properties.

The same settings can also be accessed from the camera node context menu, see *Camera node context menu on page 23*.

File group

The **File** group is used for loading and saving jobs.

Button	Description
New Job	Create a new vision job.
Load Job	Load a vision job from file into the camera.

Continues on next page

Button	Description
Save Job	Save the vision job to file.
Save Job As	

For more information on jobs and where to save them, see *Setting up a new vision job on page 51*.

Image group

The **Image** group is used for handling images.

Button	Description
Acquire Image	Get a new image.
Live Video	Turn camera live video mode on/off.
Load Image	Load an image from file into the camera.
Save Image	Save the current image to file.
Show Filmstrip	Edit record/playback settings.

Configure Job group

The **Configure Job** group is used for making the necessary settings to the current vision job. What objects the camera should locate, how to inspect the objects, and finally how the vision data should be transferred to the RAPID program.

Button	Description
Setup Image	Modify image acquisition parameters.
Calibrate	Calibrate image to real world units.
Add Part Location Tool	Add part location tool to the vision job.
Add Part Inspection Tool	Add an inspection tool to the vision job.
Output to RAPID	Select which vision job results are to be available in RAPID.
Inputs	Define camera inputs.
Outputs	Define camera outputs.
Advanced	Advanced settings and editing mode.
Run Job	Run the vision job.

For more information about configuring Integrated Vision, see *Configuring Integrated Vision on page 39*.

Production group

The **Run Mode** button in the **Production** group is used for manually switching the camera between program mode and run mode.

When running the vision system in production, switching between program mode and run mode is done from the RAPID program.

Button	Description
Run Mode	Switch between program mode (button inactive) and run mode (button active).

Continues on next page

For information on setting the run mode from the RAPID program, see *Preparing the RAPID program on page 66*.

Advanced settings

Following functionality and settings can be found under the **Advanced** button.

Button	Description
View Spreadsheet	Advanced mode editing. See section *The spreadsheet view on page 31*.
Unprotect Job / Protect Job	Protects locked cells in the spreadsheet from being edited. See section *The spreadsheet view on page 31*.
Displayed Image Settings	Configures the resolution and frame rate of live and online images. See section *Displayed Image Settings dialog on page 21*.
Job Size	Displays the size of the current job. See section *Job Size dialog on page 21*.
Update Firmware	Update the firmware of the vision camera. See section *Updating the camera firmware on page 45*.

The same settings can also be accessed from the camera node context menu, see *Camera node context menu on page 23*.

Displayed Image Settings dialog

The **Displayed Image Settings** dialog configures the resolution and frame rate of live and online images.

Setting	Description
Live Acquisition • **Resolution**	Full, half and quarter resolution for live images.
Live Acquisition • **Limit Maximum Rate**	Enables the **Maximum Rate (Frames/sec)** text box.
Live Acquisition • **Maximum Rate (Frames/sec)**	Specifies the maximum number of images the sensor may send per second (0.016 to 100). This feature can be used to reduce network traffic by limiting the number of images sent.
Online (Run mode) • **Resolution**	Full, half and quarter resolution for live images. Optimized gives half when using **Camera** or **Continuous** mode, and full when using **Manual** mode.
Online (Run mode) • **Limit Maximum Rate**	Enables the **Maximum Rate (Frames/sec)** text box.
Online (Run mode) • **Maximum Rate (Frames/sec)**	Specifies the maximum number of images the sensor may send per second (0.016 to 100). This feature can be used to reduce network traffic by limiting the number of images sent.

Job Size dialog

The **Job Size** dialog is used to configure the job size memory allocation that can be utilized by the vision system. The slider control allows to configure the memory allocation of the vision system. Adjust the slider to set the job's memory size allocation.

Continues on next page

Adjusting the **Job Size** limit of a vision system requires the vision system be rebooted, and will erase all job and settings files from non-volatile flash memory. Save or backup the job before continuing.

3.4 The controller browser

Layout of the controller browser

The **Controller** browser is a hierarchical display of controller and configuration elements as found in the **Controller** tab view in RobotStudio.

For a detailed description of the contents and functionality of the **Controller** tab view in general, see *Operating manual - RobotStudio*.

For Integrated Vision a separate **Vision System** node is available. Under this node all cameras connected to the robot appear. Named cameras are identified with their name, unnamed cameras are identified with their MAC-id.

> **Tip**
>
> When a controller is connected, the Integrated Vision add-in can be started from the context menu of the controller node, the vision system node, or the camera node in the controller browser.

Camera node context menu

On the individual camera node, a context menu is available with similar content as the **Camera** group, and the **Advanced** button on the ribbon.

Button	Description
Integrated Vision	Start the Integrated Vision add-in.
VC Configuration	Settings to find a camera on a Virtual Controller.
Connect	Connect to the selected camera.
Disconnect	Disconnect the selected camera. **Note** A camera is also disconnected when the corresponding tab in the image capture and configuration area is closed.
Rename	Rename the controller to make the camera available in RAPID with a name.
Remove	Remove the camera configuration from the controller.
Restart	Restart the camera.
Properties	View the camera properties.
Advanced	Same settings as under the **Advanced** button on the **Configure Job** group, see *Advanced settings on page 21*.

3.5 The image capture and configuration area

Layout of the image capture and configuration area

The Image capture and configuration area is used to display the image that should be processed with Integrated Vision. The image can be acquired live by any online vision camera, or be loaded from a file.

Depending on the situation, different graphical configuration guides for locating and inspecting parts, coordinate systems, and graphic data appear in the area.

Buttons in the image capture and configuration area

Button	Description
Accept changes (ENTER)	Used in some dialogs to accept a change, for example when modifying the calibration region.
Cancel changes (ESC)	Used in some dialogs to cancel a change, for example when modifying the calibration region.
Zoom In	Zoom buttons.
Zoom Out	
Zoom 1:1	
Zoom to Fit	
Zoom to Fill	
Rotate Image	Rotate the image 90 degrees.
Show Selected Tool Graphics Only	Show graphics for selected tool only.
Filmstrip	Show the filmstrip bar.
Record	Start or stop recording images to the PC.
Increase Dependency Levels	Increase the number of displayed dependency level by 1.
Decrease Dependency Levels	Decrease the number of displayed dependency levels by 1.
Reset Dependency Levels	Set the number of dependency levels displayed to 0.
Show Dependency Errors Only	Toggle on/off for displaying dependency errors only or dependency levels.

Shortcuts in the image capture and configuration area

Functionality	Shortcut
Zoom	SHIFT + scroll wheel.
Zoom using window	CTRL + left mouse button while dragging the mouse.
Pan	CTRL + SHIFT + left mouse button while dragging the mouse.

Continues on next page

Tabs in the image capture and configuration area

Each camera is displayed in a separate tab in the Image capture and configuration area. The heading of the tab displays the name of the camera together with the name of the current job.

Note

To close the tab is the same as disconnecting from the camera.

The status bar

On the bottom of the image capture and configuration area there is a status bar displaying the following information:

Information	Description
(R, G, B)@(x, y)	Syntax: (Red, Green, Blue)@(x-coordinate, y-coordinate) This information is only visible when the cursor is moved over the image area. The information shows the color and the coordinates of the pixel at the location of the cursor.
% Job Size Available	Displays the memory status of the vision camera. The percentage value shows how much memory is left to use. When the value is close to zero, all memory is consumed.
Program Mode / Run Mode	Displays the same status as the **Run Mode** button in the ribbon.

3.6 The filmstrip

Introduction

The filmstrip is used to play back images recorded to the PC or review images and results stored to the sensor.

This functionality is useful for troubleshooting and fault tracing activities. For example if there are intermittent problems of locating and inspecting parts during production.

Filmstrip settings

Click the large **Filmstrip** button on the ribbon to access the **Display and record** dialog settings window.

There are three main groups of settings, sensor settings, record settings, and playback settings.

For information about the different **Filmstrip** settings, see the *Filmstrip* section in the online **Help** tab.

The Filmstrip bar

Click the small **Filmstrip** button in the image capture and configuration area to show the filmstrip bar.

Pass images are indicated with green color, and fail images are indicated with red color.

For more information on the pass and fail status of vision tools see *Pass and fail of vision tools on page 59*.

PC filmstrip

After the filmstrip settings have been configured, and the **Record** button is pressed, the pass and fail images are recorded and viewed in the filmstrip bar. The images can be saved from an online or offline camera to any folder on the PC.

The PC playback mode can be used to play back the recorded images, the image is sent to the vision camera and the job runs against the image being played back. Up to 10,000 images can be saved to the PC.

Sensor filmstrip

When the sensor is online and acquiring images, the sensor filmstrip can be used to monitor a job's performance.

As images are acquired, the job results including the acquired image and accompanying job data are stored to the sensor's RAM. As results are stored to the sensor, a pass or fail graphic is added to the filmstrip.

When a result is highlighted in the filmstrip, the filmstrip display changes from graphics to thumbnail images and the corresponding image is loaded to the display area.

Use the controls in the **Sensor Settings** tab in the **Display and record** dialog settings window to configure the sensor's behavior.

Continues on next page

Up to 20 results can be saved, depending on the vision system's resolution and available RAM.

3.7 The palette window

Results tab

The **Results** tab displays the performance of each tool, and allows to troubleshoot or optimize the job settings.

The **Results** tab can be used to:

- Determine at a glance which tools are passing (green) or failing (red) by monitoring the semaphore of each tool.
- Visually identify the type of tool by its tool icon.
- Identify the tool by name.
- Double-click on a tool to modify it. The tool then appears in the **Context** window. Edit the parameters or graphics as needed.

It is also possible to right-click on a tool in the **Results** tab to access a short editing menu, which allows to copy, paste, delete, or edit the selected tool.

I/O tab

The I/O tab shows the active (green) or inactive (grey) status of each of the job's input and output signals. This allows monitoring of the I/O signals.

Help tab

See section *Online help on page 18*.

3.8 The context window

Introduction

The context window automatically updates to display the parameters for the selected application step. The context window is designed to guide the user through the step by presenting general parameters on the left side, and increasingly more specific parameters to the right.

In addition, depending on the tool added, the pane for the locate part and inspect part steps might also display visual aids, such as a feedback table or graph (to help configure the parameters for the application) and a range limits tab (for setting pass/fail criteria). Once a job is built, the steps can be revisited in any order, so that job parameters can be corrected and fine-tuned until the desired result is achieved.

Most of the steps require that the parameters be configured in a specific order. See the **Help** tab in the palette window for instructions for each step while developing the job.

Usage

The following functions uses the context window for settings and configurations:

- **Filmstrip**
- **Setup Image**
- **Calibrate**
- **Add Part Location Tool**
- **Add Part Inspection Tool**
- **Output to RAPID**
- **Inputs**
- **Outputs**

3.9 Options dialog

Accessing the options dialog

	Action
1	Make sure that the Integrated Vision tab is open.
2	Click the **File** tab.
3	Click **Options**.
4	Scroll to the **Integrated Vision** section.

Note

The Integrated Vision options are only visible when the Integrated Vision tab is open.

Integrated Vision options

Topic	Setting	Description
Behavior	**Immediate Feedback**	If checked, the search region in the Image capture and configuration area will be processed in *real-time* when adding and configuring vision tools. If unchecked (default), the search region in the Image capture and configuration area will be processed at *mouse click* when adding and configuring vision tools.

3.10 The spreadsheet view

Introduction

Integrated Vision inserts snippets into a spreadsheet when adding vision tools or otherwise editing the job. This is normally not shown to the user.

The spreadsheet view is an advanced mode that displays the configured job with all included vision tools in a spreadsheet mode. This is mainly intended for advanced users familiar with the *Cognex In-Sight Explorer®* software.

CAUTION

An incorrect use of the spreadsheet may result in unmanageable errors in the vision job or the RAPID program.

General

The spreadsheet is similar to other spreadsheet applications in terms of its standard operations and functionality, such as manipulating blocks of cells, editing cells, referencing cells, and inserting functions. Organized into a table of cells arranged in 400 rows (numbered 0 to 399) and 26 columns (labeled A to Z), each cell is identified by its column letter and row number. For example, cell A2 is located at the intersection of column A and row 2.

The spreadsheet is configured one cell at a time. The content of each cell is defined as a formula, and any piece of information inserted into a cell (whether a single numeric value or a complex vision processing function) is considered a part of the formula.

The spreadsheet is equivalent with the camera memory. That is when the spreadsheet is full, the camera memory is full.

For more information on using the spreadsheet see the *Cognex In-Sight Explorer®* help.

Note

A spreadsheet can contain a maximum of 4,096 active cells. Inserting a function (such as *FindBlobs*) that, in turn, inserts multiple vision data access functions into the spreadsheet can cause the spreadsheet to attempt to exceed the limit. In this case, no warning dialog is displayed. The cells over the 4,096 limit are simply not inserted.

Transfer data to RAPID

Any data that gets updated with each image acquisition can be transferred from the spreadsheet to RAPID using the **Output to RAPID** table. To make the data visible in the **Output to RAPID** table, a **Symbolic Tag** must be added to the cell holding the data.

The name of the tag must have the format *<Group>.<Result>*, for example `mydata.data`.

The **Symbolic Tag** is added from the right-click menu.

Continues on next page

Changing the language

When changing the application language for the GUI in RobotStudio, the GUI language of the Integrated Vision add-in will also change.

The integrated *Cognex EasyBuilder®* software creates new jobs in the currently set GUI language and populates the cells of the spreadsheet with translated names. This results in that when the job is opened using another GUI language setting, a lot of the data will have names in the language that was set when first creating the job. Any subsequent additions are created in the currently set language.

For this reason, it is highly recommended to use the same GUI language while programming and using the job.

Even though some labels describing data and settings are created in the local language, RobotStudio is set to create data with names (symbolic tags) in English. The reason is that data which may be accessed from RAPID shall not require language encoding, for example when using `CamGetParameter`.

Note

It is not recommended to change the RobotStudio language after the configuration of an Integrated Vision system has started.

Shortcuts in the spreadsheet view

Shortcut	Functionality
CTRL + 1	Open the **Format Cells** dialog

Right-click menu

Right-clicking within the spreadsheet displays a menu that allows various spreadsheet operations to be performed:

Setting	Description
Cut	Cuts the selected cell(s).
Copy	Copies the selected cell(s) to the clipboard.
Paste	Pastes cell(s) that were cut or copied to the clipboard.
Insert	Inserts rows or columns equal to the number of rows or columns selected. If entire rows or columns are not selected, the Insert dialog opens to insert cells, rows or columns.
Delete	Deletes the selected rows or columns. If entire rows or columns are not selected, the **Delete** dialog opens to delete cells, rows or columns.
Clear Contents	Clears the contents of the active cell(s).
Insert Function	Opens the dialog to insert a function into the active cell. This option is available only if the active cell does not contain a function.
Edit Function	Opens the property sheet if the active cell contains a function with an associated property sheet, otherwise opens the **Insert Function** dialog. This option is available only if the active cell contains a function.
Insert Absolute Reference	Inserts an absolute cell reference into the active cell.

Continues on next page

Setting	Description
Insert Relative Reference	Inserts a relative cell reference into the active cell.
Insert Comment	Opens the dialog to insert a new comment for the active cell. This option is available only if the active cell does not contain a comment.
Edit Comment	Opens the dialog to edit an existing comment for the active cell. This option is available only if the active cell contains a comment.
Insert Symbolic Tag	Opens the symbolic tag editor to insert a new symbolic tag for the active cell. This option is available only if the active cell does not contain a symbolic tag.
Edit Symbolic Tag	Opens the symbolic tag editor to edit an existing symbolic tag for the active cell. This option is available only if the active cell contains a symbolic tag.
Cell Graphic	Enters interactive graphics mode to edit the cell graphic. This option is available only when the active cell contains a function that has an associated cell graphic.
Cell State	Opens the dialog to enable or disable the execution of cells either explicitly or conditionally, based upon the value of a referenced cell.
Set Job Pass/Fail	Opens the setup dialog to monitor a cell, which contains a numeric result, to determine the job's overall pass/fail status. The job status can be sent to the RAPID program.
Custom View Settings	Opens the dialog to configure the properties of the custom view of the spreadsheet.
EasyView Settings	Opens the dialog to customize how data is displayed.
Format	• **Format Cells**: Opens the dialog to format the number, alignment, font and protection of the selected cell(s). • **Row Height**: Opens the dialog to adjust the height of one or more spreadsheet rows. • **Column Width**: Opens the dialog to adjust the width of one or more spreadsheet columns. • **Hide**: Hides the selected row(s) or column(s). • **Unhide**: Reveals the hidden row(s) or column(s) that lie between the selected row(s) or column(s).
Snippet	• **Import**: Opens the dialog to import the snippet, saved as a .CXD file, into the spreadsheet. This data file can be loaded from the snippets folder on the PC. • **Export**: Opens the dialog to export the snippet, saved as a .CXD file, to the snippets folder on the PC.
Import Cells	Imports cell data as a .CXD file into the spreadsheet.
Export Cells	Exports cell data as a .CXD file.

Unprotect Job / Protect Job dialog

The protect job dialog prevents the modification of all locked cells in the job. This can be useful for jobs with cells that must remain unchanged for the job to function correctly, but allow the rest of the job to be customized.

A job is protected with a user defined password, which can remain blank. The password is required to unprotect the job.

When the job is protected, formulas within any locked cell cannot be modified, but the formatting of the locked cells can still be modified (font, color, etc.). The formula

Continues on next page

of a locked cell remains visible, but grayed out, when the mouse pointer is resting on the locked cell.

Note

Use the **Protection** tab in the **Format Cells** dialog to lock cells.

4 The FlexPendant user interface

4.1 RobotWare Integrated Vision

Introduction

This section gives an overview of the graphical user interface of the Integrated Vision application on the FlexPendant. Views, buttons, and other parts of the user interface are described in respect of their content and how they are accessed.

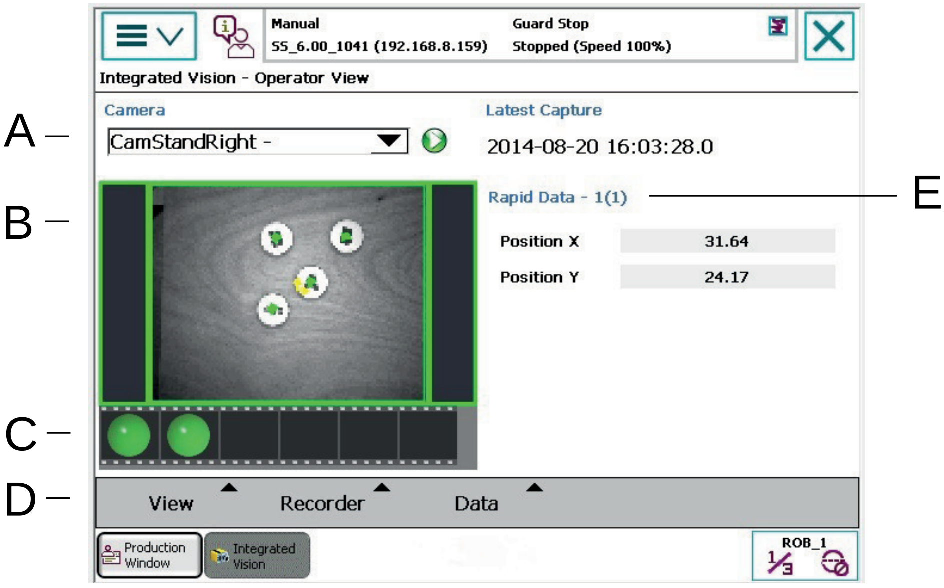
For more information on how to use the FlexPendant in general, see *Operating manual - IRC5 with FlexPendant*.

Use this procedure to start RobotWare Integrated Vision.

	Action
1	Tap the **ABB** menu.
2	Tap **Integrated Vision**.

4.2 Operator view

The Operator View interface



xx1200000987

	Parts	Description
A	**Camera**	Selects connected camera and displays active job.
B	Image area	Displays an image acquired by the vision camera.
C	Recorder bar	Used to record a sequence of images for later analysis.
D	**View**	Used to change the view to show either image, results, or both.
	Recorder	Used to show and freeze the recorder bar and to save images.
	Data	Used to setup the results view.
E	Results area	Used to display camera results or RAPID data. User configured.

View settings

The default setting is to display image and results side by side, but it is also possible to only view either the image or the results.

Setting	Description
Image & Results	Displays image and results side by side.
Image	Displays the image only.
Results	Displays the results only.

Tip

Tap the image area to go to full screen mode. Full screen mode is active for 30 seconds, then it is automatically disabled.

Continues on next page

Recorder settings

The recorder bar on the FlexPendant is a simplified version of the filmstrip bar in RobotStudio, see *The filmstrip on page 26*.

Pass images are indicated with green color, and fail images are indicated with red color.

Setting	Description
Show	Shows the recorder bar.
Freeze	Freezes the recorder bar. When the recorder bar is frozen it is possible to view the individual images by tapping on them.
Save	Save can only be performed when the recorder bar is frozen. The active image is saved as a *.bmp* file with the current timestamp to the *"...\HOME\IV"* folder of the controller flash disk.

Data settings

The results area can be customized to display any RAPID data that is of interest to the user. The layout is job specific, so a new layout needs to be created for each job. It is also possible to configure a default layout that can be displayed when no job is loaded.

Note

Only RAPID data declared as persistent (`PERS`) can be displayed in the results area.

Setting	Description
Configure	Configure the layout of the results area for the job by adding labels and RAPID data.
Camera Results	Displays the vision camera parameters that are mapped in the **Output to RAPID** dialog. See *Output to RAPID on page 60*.
Default	Displays the default layout of the results area when no job is loaded.

The configurations are saved in the *IVSetup.xml* file found in the *"...\HOME\IV"* folder on the controller flash disk. Users familiar with xml-files can edit the *IVSetup.xml* file directly instead of using the configuration tool on the FlexPendant. Restart the RobotWare Integrated Vision application on the FlexPendant to apply the changes.

This page is intentionally left blank

5 Configuring Integrated Vision

5.1 Recommended working procedure

General

This section describes the recommended working procedure when creating a new vision application. The working procedure helps to understand the dependencies between the different objects. A good approach when creating a new application is to start with the basic functionality. When that works as expected, expand the application.

A prerequisite is that all steps in the hardware and software installation procedures must have been performed, see chapter *Installation on page 15*.

Basic steps

Use this procedure to create a new vision application.

	Action	See
1	Make some initial preparations.	*Preparations on page 40*
2	Setup the camera.	*Setting up the camera on page 41*
3	Create a new vision job.	*Setting up a new vision job on page 51*
4	Adjust the image settings of the vision camera.	*Setting up the image on page 52*
5	Calibrate the camera and the robot.	*Calibration on page 54* *Calibration theory on page 76*
6	Add vision tools to locate and inspect parts in the image.	*Adding vision tools on page 57*
7	Make the vision data available to the RAPID program.	*Output to RAPID on page 60*
8	Setup the inputs and outputs of the vision camera, if any.	*I/O handling on page 64*
9	Prepare the RAPID program on the controller.	*Preparing the RAPID program on page 66* *Technical reference manual - RAPID Instructions, Functions and Data types*
10	Start production.	*Starting production on page 71*

For more useful tips about setting up a vision system, see *Best practise on page 79*.

5.2 Preparations

Preparations

Experience shows that when starting with a clean system it is good to first load a RAPID program and make some initial preparations.

- Create tool data for all needed tools, and define the TCPs.
- Create work object data for all needed fixtures and define them.
- Etc.

The recommendation is to create a module and add the `MoveToDetectedObject` snippet. This way all the data that is edited during calibration, grip point training etc. will be in place.

Tip

Use the `MoveToDetectedObject` snippet as a base when creating a new vision program.

For more information on adding snippets, see *RAPID snippets in RobotStudio on page 66*.

5.3 Setting up the camera

5.3.1 Basic procedures

Configuring the camera network and connecting to a camera

When all cameras are physically connected, each camera needs to be configured with an IP-address and a name.

The IP-address for the camera is by default assigned automatically by the robot controller, using DHCP, but it is also possible to set a static IP-address.

The camera name is used as a unique identifier for the camera in all parts of the system, for example RobotStudio, RAPID programs etc. This enables that the IP-address of the camera can be changed, for example if the camera is replaced, without having to modify the program.

The controller browser in RobotStudio has a node called **Vision System**. This is used for configuring and connecting to cameras. A connection to a camera is established through the robot controller. The camera is connected as a FTP remote mounted disk.

The configurations of the cameras are stored in the system parameters of the controller. For more information about these system parameters, refer to the topic Communication - type Application Protocol in *Technical reference manual - System parameters*.

Use this procedure to assign cameras to the controller:

	Action
1	Make sure that the network adapter of the PC is set to obtain an IP-address automatically.
2	Make sure that any installed firewalls on the PC allows communication with the camera, see *Firewall settings on page 43*.
3	Connect the PC to the MGMT port (management port) on the controller.
4	Start RobotStudio.
5	Connect to the controller and request write access over the controller.
6	Go to the **Controller** tab on the ribbon menu and start the Integrated Vision add-in.
7	Go to the **Vision** tab on the ribbon menu.
8	Expand the **Vision System** node of the controller browser. Named cameras are displayed with their name, unnamed cameras are displayed with their MAC-id. **Note** If the camera does not show in the **Vision System** list, the IP-address of the camera can be set to a different subnet. For more information, see *Connecting to a camera on a different subnet on page 44*.
9	Right-click on the camera and select **Connect**. The image from the camera should now appear in a separate tab in the Image capture and configuration area. Use the camera image to identify the correct camera.
10	If necessary, update the image by pressing the **Acquire Image** button.

Continues on next page

	Action
11	Right-click on the camera and select **Rename**.
12	In the **Rename** dialog, enter the name of the camera in the **RAPID Camera Name** field. **Note** It is advisable to also set the camera **Host Name** to the same name as the **RAPID Camera Name**.
13	Restart the controller and the camera. **Note** It is important that both the controller and the camera is restarted.
14	The configured camera should now appear in the **Vision System** node of the controller browser.

Note

The names of the configured cameras are stored in the system parameters of the controller, topic **Communication** (`SIO.cfg`). The **IP**-settings are stored in the cameras.

Disconnecting a camera

To disconnect from a camera, click **Disconnect**, or simply close the corresponding tab in the image capture and configuration area.

Removing a camera

To remove a configured camera, click **Remove** in the camera node context menu. The camera can also be deleted in the system parameters of the controller, topic **Communication** (`SIO.cfg`).

Use this procedure to remove a camera:

	Action
1	Use the **Configuration Editor** in RobotStudio, or an offline editor, to open the system parameters of the controller, topic **Communication** (`SIO.cfg`).

2	Delete the application protocol for the selected camera:
	`IV_CAMERA:`
	`-Name "MyCamera" -Type "CAMERA" -Trp "TCPIP1" -MAC "..."`

3	Restart the controller.

For more information about using the **Configuration Editor** in RobotStudio, see *Operating manual - RobotStudio*.

Continues on next page

Firewall settings

The following table shows the necessary firewall configurations for Integrated Vision:

Name	Action	Direction	Protocol	Remote Address	Local Service	Remote Service	Application
Telnet	Allow	Out	TCP/IP	Any	Any	23	RobotStudio.exe
In-Sight Protocol	Allow	Out	TCP/IP	Any	Any	1069	RobotStudio.exe
In-Sight Discovery	Allow	In/Out	UDP/IP	Any	1069	1069	RobotStudio.exe
DataChannel	Allow	Out	TCP/IP	Any	Any	50000	RobotStudio.exe
Upgrade port (PC only)	Allow	Out	TCP/IP	Any	Any	1212	RobotStudio.exe

5.3.2 Additional camera configuration

Changing the IP-address of a camera

The IP-address of the camera can be changed from the **Network Settings** dialog.

It is advisable to set the camera **Host Name** to the same name as the **RAPID Camera Name** given in the **Configuration** dialog. The system works even if this is not the case, but some of the integrated *Cognex EasyBuilder®* dialogs in RobotStudio use the host name to identify the camera.

Use this procedure to change the network setting and the camera host name:

	Action
1	Select the camera in the **Vision System** node of the controller browser.
2	Click on the **Connect** button drop-down menu and select **Network Settings**.
3	Either set a fixed IP-address on the same subnet as the controller and the PC, or enable DHCP.
4	It is advisable to also set the camera **Host Name** to the same name as the **RAPID Camera Name** given in the **Configuration** dialog. **Note** Do not change the **Telnet Port** or any other settings.
5	Click **Ok**.
6	Restart the camera.
7	Restart the controller.

Connecting to a camera on a different subnet

If the IP-address of the camera is not on the same subnet as the controller and the PC, the camera will not show in the **Vision System** node of the controller browser. Therefore it is not possible to set a new IP-address using the **Network Settings** dialog.

Use this procedure to connect to cameras on a different subnet:

	Action
1	Click on the **Connect** button drop-down menu and select **Add Sensor**.
2	Click the camera in the list and either set a fixed IP-address on the same subnet as the controller and the PC, or enable DHCP.
3	Click **Apply**.
4	Restart the camera.
5	Restart the controller.

Changing the date and time of a camera

Use this procedure to change the date and time of a camera:

	Action
1	Select the camera in the **Vision System** node of the controller browser.
2	Click on the **Connect** button drop-down menu and select **Set Date and Time**.

Continues on next page

	Action
3	Change the date and time settings.
4	Click **Ok**.
5	Restart the camera.

Updating the camera firmware

When using a vision camera that was not delivered together with the Integrated Vision option, it may be necessary to update the camera firmware.

The latest firmware for the *Cognex In-Sight®* cameras is included in the RobotStudio installation.

Use this procedure to update the camera firmware:

	Action
1	Select or right-click the camera in the **Vision System** node of the controller browser.
2	Click on the **Advanced** drop-down menu and select **Update Firmware**. The camera model, the current version, and the new version of firmware is displayed.
3	Click **Update**.
4	Restart the camera.
5	Restart the controller.

Note

When running Integrated Vision, it is recommended that the camera has the following firmware version:

- DSQC1020, DSQC1021, and In-Sight Micro 1402: 4.10.05
- DSQC1063 and DSQC1064: 5.07.03 or 6.2.1
- DSQC1098 and DSQC1099: 6.2.1
- AE3 Smart Gripper: 4.09.04

Connecting to a camera on a virtual controller

Connect to a camera on a virtual controller if a real controller is unavailable. The procedure to connect to a camera on a virtual controller is the same as connecting to a camera on a real controller. Before connecting to the camera, ensure that the service port is connected to the correct network so that the virtual controller listens to it.

	Action
1	Check the IP-address of the PC network adapter to which the camera is connected.
2	Right-click the **Vision System** node and select **VC Configuration**.
3	Select the IP address (adapter) to which the camera is connected.

Continues on next page

Limitations

Note

- The virtual controller does not have the same performance as a real controller when communicating with the camera.
- The virtual controller is provided "as is", therefore it should only be used for evaluation, training, support, testing, and similar.

Connecting to a camera emulator on a virtual controller

A camera emulator is available when creating a system with a virtual controller. The emulator model is 7905 and the firmware is 6.02.01. When the emulator is used to view and image, the image is cropped to the size of the emulated sensor.

In the emulator, it is always the currently loaded image that is evaluated. The emulator has no functionality to loop through images.

The vision tools inside a camera emulator does not support the `CamReqImage` instruction. The vision job must be evaluated and tested using the RobotStudio user interface.

To test and evaluate the RAPID program that includes the `CamReqImage` instruction, the values in the **Output to RAPID** must remain constant and these values should not be linked with the vision tools. To achieve this, create one or several parts in the **Output to RAPID** and add constant values to the variables x and y.

The emulator is started by right clicking the Vision System tab in RobotStudio and select **Start emulator**. It will be connected to the loopback interface on the PC, 127.0.0.1. In the output window in RobotStudio, messages are displayed when the emulator executable is started and when the emulator is added to the Vision System tab. Select **Refresh** in the Vision System tab to add the emulator to the virtual controller. Rename the emulator in the same way as a real camera.

Limitations

- The camera emulator is provided "as is", therefore it should only be used for evaluation, training, testing, and similar.
- The camera emulator is only accessible when using a virtual controller.
- It is only possible to have one emulator added to the virtual controller.
- The camera emulator is of model 7905 and with firmware 6.02.01. It is not possible to change model or firmware of the emulator.
- It is not possible to set a host name for the camera emulator.
- The camera emulator does not allow the controller to acquire an image. Hence, a warning message gets displayed while attempting to run the `CamReqImage` instruction against a camera emulator.

5.3.3 Restricting user access

Introduction

The Integrated Vision add-in in RobotStudio communicates directly with the camera. For that reason the camera itself has got a user authentication method to provide the possibility to restrict the use of certain functionality.

RobotStudio and the controller both use a password to log on to the camera. By default the user name is *"admin"* and the password is blank *""*.

It is possible to edit the list of users on the camera and their associated privileges, and also to select which user name and password to be used the controller when it communicates with the camera. When updating the user list on the camera, make sure to also update the user profile used by the robot controller by clicking **Set controller user**.

User authorization system (UAS)

The user authentication method of the camera is separate from the user authorization system (UAS) of the robot controller.

Some actions specifically related to the controller requires write access, such as naming the camera. This is since the information is stored in the controller configuration. For all other actions, the user authentication of the camera applies.

For more information about UAS, see *Operating manual - RobotStudio*.

Editing user access settings

The **User Access Settings** dialog maintains the access level and FTP read/write privileges for authorized users of *Cognex In-Sight®* vision systems. The **User Access Settings** dialog determine which users may log onto a particular camera, as well as the types of changes they can make to the active job. Each camera has its own user list, separate from every other camera on the network. If a user needs access to a particular camera, they must know a user name and password that already exists in that camera's user list.

Note

Every camera is pre-configured with three users: *admin*, *operator*, and *monitor*. These users are configured for *Full*, *Protected*, and *Locked* access levels, respectively.

Note

The maximum number of users that can be added to one camera is 32.

Setting the controller user

Use this procedure to select the user profile to be used by the controller when communicating with the camera.

	Action
1	Request write access over the controller.

Continues on next page

	Action
2	Click on the **Connect** button drop-down menu and select **Set Controller user**.
3	Click on the **Select User** drop-down list and select a user.
4	Click **OK**.
5	Restart the controller.

Adding a new user

Use this procedure to add a new user.

	Action
1	Click on the **Connect** button drop-down menu and select **User Access Settings**.
2	Click **Add**.
3	Enter an alphanumeric string for the **User Name** field. **Note** The length of the user name and password strings cannot exceed 30 characters, and both are case-sensitive.
4	Enter a password for the new user.
5	Click on the **Access** drop-down list and select an access level for the new user. For more information, see *Access levels on page 48*.
6	Optionally, disable the **Allow Run Mode**/**Program Mode** checkbox to restrict users with *Protected* access from toggling the program mode/run mode state of the sensor. This checkbox is grayed out when the selected access level is *Full* or *Locked*.
7	Optionally, enable the **Allow Run Mode Job Save** checkbox to allow users with *Full* or *Protected* access to save jobs while in run mode. This checkbox is grayed out when the selected access level is *Locked*. **Note** When checked, users with *Protected* access can save jobs in run mode even if the FTP write privilege is not enabled.
8	Specify the user's FTP, read, and write privileges. For more information, see *FTP privileges on page 49*.
9	Click **OK**.

Access levels

The access level controls how much interaction is allowed for the current user to prevent inadvertent or unauthorized changes to the configuration. The selected access level will be in effect whenever anyone logs on to the RobotStudio vision add-in with the selected user name and password.

Three access levels are available:

Access level	Description
Full	This offers complete, unrestricted access to the camera. Any job created in RobotStudio can be loaded, modified or saved, and all menu selections are available. The default *admin* user account has full access.

Continues on next page

Access level	Description
Protected	The user has limited access to the sensor. Protected mode allows you to access *Live Video* mode, toggle the *Run Mode/Program Mode* status of the camera (if permitted by your allow run mode/program mode privileges), and open or save jobs (if permitted by your FTP read/write privileges). In the standard view (not spreadsheet view), protected access allows you to edit tool parameters, but not to add or delete tools. In spreadsheet view, protected access always displays the custom view of the spreadsheet. A user in protected mode can edit the values of any graphics controls functions visible in the custom view, but cannot change the functions themselves. The default operator user account has protected access. For more information about the spreadsheet view and the custom view, please refer to *Cognex In-Sight® Explorer User's Guide* available on the Cognex web-site.
Locked	This offers the most restrictive access. You can only monitor the operation of the current camera. The default monitor user account has locked access.

FTP privileges

FTP read privileges apply to opening job or image files from the camera, while FTP write privileges apply to saving jobs or images to the camera.

Additionally, these permissions are in effect when the user attempts to log on to the active camera from a remote FTP client on the network. More specifically, these privileges need to be enabled for the controller to be allowed to carry out file operations on the camera.

If any firewall blocks the communication between the PC and the camera during a save job or load job operation, a network error gets displayed. For information on how to configure the firewall, see chapter *Getting Started*, section *Network settings* in *Operating manual - RobotStudio*.

Privileges and access levels needed by the controller user

The user profile used by the robot needs FTP read/write privileges to allow file transfer between the camera and the controller.

The required access level is **Protected**, since this is the lowest level that allows changing between run mode and program mode. Make sure that **Allow Run Mode**/**Program Mode** is checked.

> **Note**
>
> The user profile and associated password used by the controller to log onto the camera is stored in plain text in the robot configuration. Keep the privileges for this user profile to the necessary minimum.

Editing an existing user

Use this procedure to edit an existing user.

	Action
1	Click on the **Connect** button drop-down menu and select **User Access Settings**.
2	Click **Edit**.

Continues on next page

	Action
3	Modify the user settings as necessary. For more information about the settings see *Adding a new user on page 48*.

Note

- If the *admin* password is changed, you must log off then log back on to the sensor with the new password or errors may occur.
- The access level, allow run mode/program mode privileges, FTP privileges, and user name for *admin* cannot be modified.
- The **Show Custom View at Log On** checkbox affects only the spreadsheet view. The standard RobotStudio view does not support custom views.

Deleting an existing user

Use this procedure to delete an existing user.

	Action
1	Click on the **Connect** button drop-down menu and select **User Access Settings**.
2	Click **Delete**.
3	Confirm the deletion.

Note

The user *admin* cannot be deleted.

5.4 Setting up a new vision job

Connecting to the camera

Open RobotStudio and make sure that the camera is connected and tested according to the instructions in *Configuring the camera network and connecting to a camera on page 41*.

Creating and saving a new job

All camera configurations and settings in Integrated Vision put together, is called a *job*.

The active job is stored in the working memory of the camera and is lost in the case of a power fail. The job shall be permanently stored as a job file (.job) either on the flash disk of the camera or on the flash disk of the robot controller.

To be able to load the job file from the RAPID vision instructions, the job must be stored on the camera flash disk, but it is strongly recommended to also create a backup elsewhere in case the camera gets damaged.

If the robot controller is used for job storage, then the job must first be copied from the controller disk to the camera flash disk before loading the job into the camera memory. This is done with standard RAPID file handling instructions.

Tip

For file transfer examples see the backup and restore snippets in RobotStudio.

Use this procedure to create a new job.

	Action
1	Make sure that the camera is in **Program Mode**.
2	Click **New Job** in the ribbon.
3	Click **Yes** to clear all data from the current job.
4	Click **Save Job** or **Save Job As** in the ribbon. The **Save As** dialogue will appear since the job has not been saved before.
5	Browse to the desired location, preferably on the camera flash disk.
6	Name the job and click **Save**. The name of the job will appear on the image tab in the image capture and configuration area.

Position the camera

If the camera is mounted on a moving part of the robot, now is the time to jog the camera into position and to store this position.

Before acquiring a new image, the camera always needs to be brought back to exactly the same position to maintain accuracy.

5.5 Setting up the image

Introduction

The most commonly used setting is the exposure time. A longer time allows more light into the camera and makes the image brighter.

Adjusting the settings of a vision job is often an iterative process and the exposure time often needs to be modified once or more before the job is ready. Sometimes the setting that is required for obtaining a clear image for calibration may not be exactly the same as the optimal setting for detecting the product. If a subsequent step proves that the image settings are not optimal, do not hesitate to go back a modify them.

Tip

For more information see the *Best practise on page 79* chapter. Especially sections *Obtain good lighting on page 84* and *How to mount the camera on page 81*.

Image trigger

The image trigger setting decides which event that triggers the camera to acquire an image. To be able to trig the image from the Integrated Vision RAPID instructions, the trigger setting must be set to **Camera** or **External**.

Note

If the RAPID instruction `CamReqImage` is used, set the camera image trigger type to **Camera**. If the same RAPID instruction, `CamReqImage`, is used with the optional argument `\AwaitComplete`, then the camera image trigger type has to be set to **External**.

Image settings

Use this procedure to setup the image.

	Action
1	Make sure that the camera is in **Program Mode**.
2	Click **Setup Image** in the ribbon.
3	In the **Context** window, change the **Trigger** setting to **Camera** or **External** if using `CamReqImage` with optional argument `\AwaitComplete`.
4	If necessary, adjust the other settings to get the best image quality.
5	Save the job.

For information about all parameters and settings, see the *Set Up Image / Adjusting the Edit Acquisition Settings* section in the online **Help** tab.

Continues on next page

Note

The image settings are connected directly to the camera memory. All changes are applied immediately. Therefore there are no apply, save or undo buttons available. Save and backup the job frequently to avoid loss of data.

5.6 Calibration

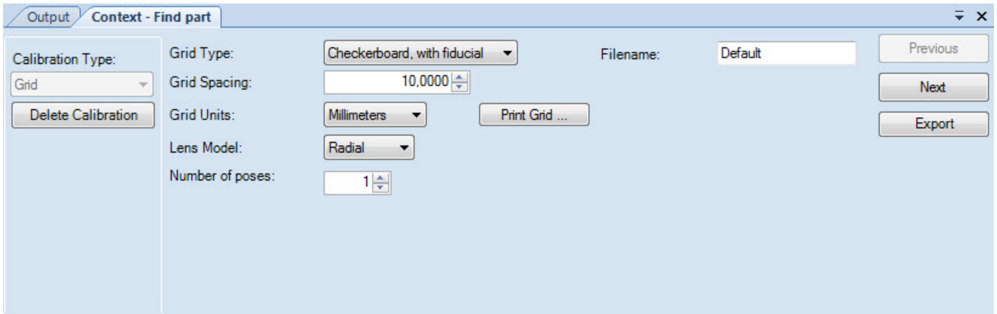
Introduction

An image consists of pixels, so in order to get the result in mm the camera needs to be calibrated. The **Calibrate** function is used to calibrate the image to real-world units.

The calibration consists of two basic steps. First the camera calibration which converts the image pixels to mm, and second the camera to robot calibration which relates the camera coordinates to a robot frame (work object).

The settings are displayed in the **Context** window. The different camera calibration types are described in the online **Help** tab, and in section *Calibration theory on page 76*.

To get the best accuracy the recommended calibration type for Integrated Vision is to use a checkerboard with fiducial, see *Camera to robot calibration on page 55*. The fiducial mark provides a clear reference which is later used for defining a corresponding work object.



xx1300001097

> **Note**
>
> It is outmost important to setup the right size of the calibration grid, otherwise the calibration will be wrong.

Camera calibration

Use this procedure to calibrate the camera.

	Action
1	Make sure that the camera is in **Program Mode**.
2	Click **Calibrate** in the ribbon.
3	In the **Context** window, change the **Calibration Type** to **Grid**.
4	From the **Grid Type** drop-down menu select one of the checkerboard calibration plates with fiducial, reference point.

Continues on next page

	Action
5	If necessary, adjust the spacing, units, lens model, and number of poses settings. • Use mm as the unit • The lens model depends on from where the most distortion is expected. Either because the camera is viewing from an angle (projection), or that the lens itself is distorting the image (radial). • Number of poses allows to use more than one image of the calibration plate to calibrate the camera in case the plate does not cover the full field of view.
6	Click **Print Grid** to print the calibration plate. The printed image must have a high contrast and the paper must not be reflective (high gloss). Verify with a ruler that the squares are proportional.
7	Place the calibration plate on a fixed position in the center of the camera image, at the same height as the objects that the camera shall identify. The calibration paper must be completely flat, adequately illuminated, and free from gloss and shadows. Rotate the calibration plate so that the X and Y arrows corresponds to the desired direction of the work object.
8	In the **Context** window, click **Next**. The calibration is now being calculated by the camera, and the number of found feature points are displayed.
9	Click **Next**.
10	Click **Calibrate** to apply the calibration.
11	Click **Finish** to complete the calibration.
12	Save the job.
13	Do not change the position of the calibration plate until the work object has been defined.

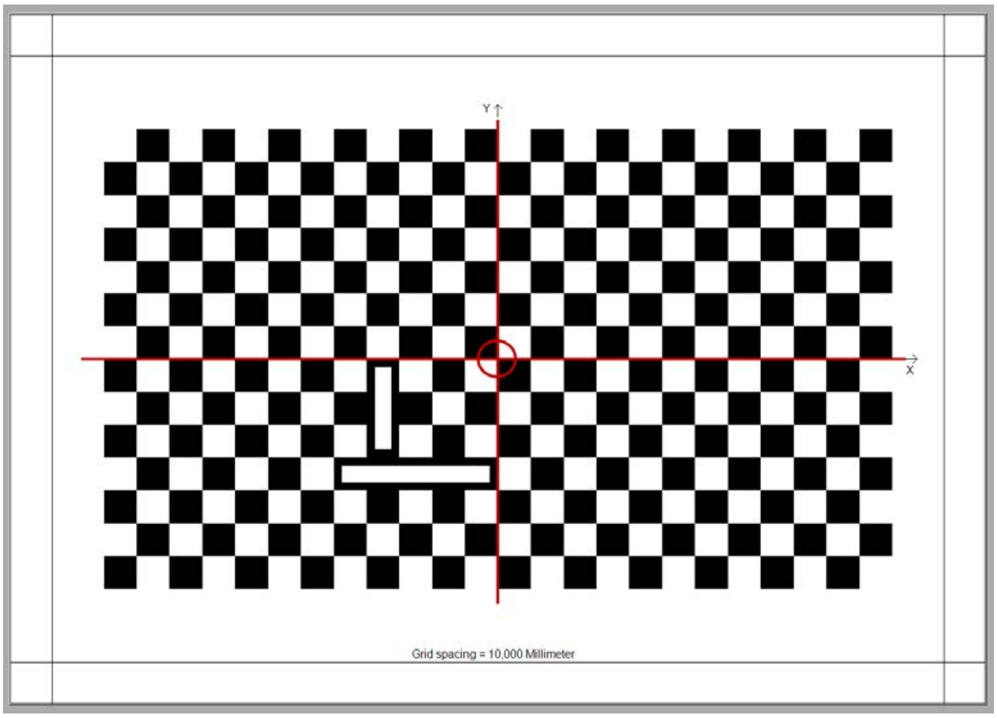
For more information about parameters and settings, see the *Set Up Image / Calibrating the Image to Real World Units* section in the online **Help** tab.

Camera to robot calibration

The camera is calibrated to the robot by defining a work object with the same origin of coordinates as the calibration plate.

On a checkerboard calibration plate with fiducial, the origin of coordinates is located at the intersection of the extended X- and Y-arrows as seen in the picture below.

Continues on next page



xx1200000994

Note

Before defining the work object, make sure to check where the origin is located on the currently used calibration plate.

Use this procedure to calibrate the robot.

	Action
1	Create a pointing tool and define the tool TCP.
2	Create one work object for each camera.
3	Activate the pointing tool and define the user frame of the work object along the corresponding x- and y-axes of the calibration plate. Leave the object frame empty.
4	Test that the calibration is correct by jogging the robot in the work object.
5	The calibration plate can now be removed.

For information on how to create and define tools and work objects, see *Programming and testing* in *Operating manual - IRC5 with FlexPendant*.

5.7 Adding vision tools

Introduction

The location tools are used to define a feature in the image that provides positional data. The location tools creates a reference point, which is used to locate a part in the image quickly and reliably, even if the part being inspected rotates or appears in different locations in the image.

The inspection tools are used to examine the parts located by the location tool. Depending on the requirements of the current application different tools are available for checking presence/absence, measurement, counting, geometry, etc.

The settings are displayed in the **Context** window.

Adding a location tool

Use this procedure to add a location tool.

	Action
1	Load or acquire a new image. Make sure that the part to be located is present within the areas in the image where the part may appear.
2	Click **Add Part Location Tool**, then click the desired tool from the drop-down menu.
3	Follow the tool specific instructions in the context window.
4	If necessary, adjust the settings to get the best performance.
5	Save the job.

For information about all parameters and settings, see the *Locate Part* section in the online **Help** tab.

Most commonly used location tools

Part location tool	Description
PatMax® Pattern **PatMax® Patterns (1-10)**	Locates a single pattern, or up to ten patterns using the `PatMax®` algorithms, and reports the x- and y-coordinates, angle and score of the found patterns.
Blob **Blobs (1-10)**	Locates a single group, or up to ten groups, of dark or light-colored connected pixels, called blobs, and reports the x- and y-coordinates of the found blob. This tool is commonly used as a fixture to orient other vision tools.

Most commonly used location tool settings

Setting	Description
Number To Find	Defines the number of instances to detect. The default value is often 1 and must be increased to detect multiple instances.

Continues on next page

Setting	Description
Rotation Tolerance	Defines how far the the found pattern can be rotated from the trained pattern and still be recognized as a valid pattern. The default value of +/-10° to 15° is often too small and needs to be increased. **Note** The `PatMax®` Vision tool consists of `PatQuick` and `PatMax` steps, rotation tolerance setting applies only to the `PatQuick` step. Hence, the final rotational result of the part might be slightly higher than the expected value. To obtain an exact rotation limit, a limit check in the RAPID code is recommended.

Adding an inspection tool

Use this procedure to add an inspection tool.

	Action
1	Load or acquire a new image. Make sure that the pattern to be located is present within the areas in the image where the pattern may appear.
2	Click **Add Part Inspection Tool**, then click the desired tool from the drop-down menu.
3	Follow the tool specific instructions in the context window.
4	If necessary, adjust the settings to get the best performance.
5	Save the job.

For information about all parameters and settings, see the *Inspect Part* section in the online **Help** tab.

Most commonly used inspection tools

Tool group and part inspection tool	Description
Presence/Absence Tools • **PatMax® Pattern**	Determines whether or not a trained pattern is present or absent, using the `PatMax®` algorithm. Reports a pass if the pattern is present and within limits, or a fail if it is outside of the limits.
Presence/Absence Tools • **Blob** **Blobs (1-10)**	Determines whether or not a group of dark or light-colored connected pixels, called blobs, are present or absent. Reports a pass if the blob feature is present and within limits, or a fail if it is outside of the limits.
Measurement Tools • **Distance**	Measures the distance between any two features such as edges, circles, patterns, and/or blobs. Reports a pass and the distance in millimeters or pixels (unless the image is calibrated), or a fail if the reported distance is outside of the limits.
Identification Tools • **PatMax® Patterns (1-10)**	Determines from a library of trained patterns which pattern best matches the pattern in the image, using the `PatMax®` algorithm. Reports the name of the found pattern and it's score compared to the trained model and results in a pass if the found pattern falls within limits, or a fail if it is outside of the limits or the pattern wasn't found.

Continues on next page

Pass and fail of vision tools

For each location and inspection tool there is a check box that defines whether or not the tool's pass/failure status should be included in the job's overall pass/fail status. By default, it is checked and will be included in the job's overall pass/fail status.

Uncheck the check box to keep the tool's pass/failure status separate from the other tool's in the job.

This control should be unchecked if a tool is expected to fail. For example, if two identification tools were being used to determine whether or not the part was a right-hand or a left-hand part. One of the tools would be expected to fail every time. If the checkbox was checked, this situation would result in the job failing every time regardless of which side of the part was identified.

Links between vision tools

The output results of one vision tool can be used as input parameters for another tool. Then the tools are linked.

Tool properties are divided into input and output properties, and links can only be established from output properties to input properties. Output properties can be linked to multiple separate input properties, while an input property only accepts one output property linked into it. The most common example is when a location tool is used as a fixture for the inspection tool.

The links are displayed visually as graphical arrows in the **Results** tab. The graphical arrows cannot be edited. To remove or change a link, the parameters has to be removed from the vision tool.

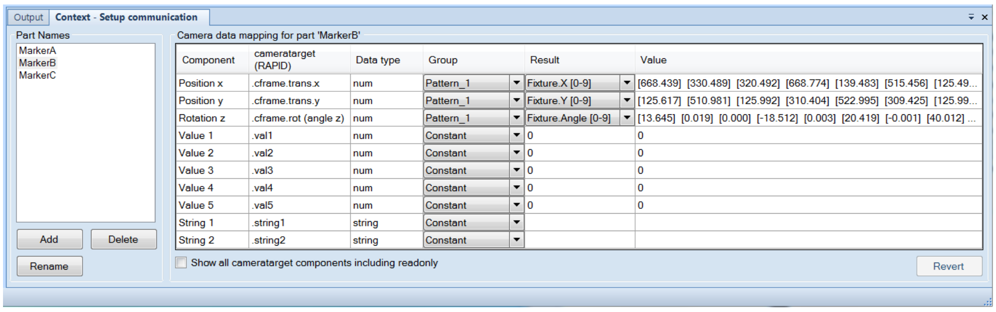
5.8 Output to RAPID

Introduction

The camera job produces a number of parameters with each image that is acquired. Most important is the output of the configured vision tools, but also additional data such as the used exposure time etc.

The mapping dialog in **Output to RAPID** provides a simple but flexible way for the user to select which data to be converted to RAPID variables. The purpose is to allow the user to experiment and alter the vision job without having to modify the RAPID program that makes use of the vision data.

The dialog allows the user to click and select which vision output parameter to be mapped to a specific property of the RAPID `cameratarget` record. It also allows creating categories of camera targets by defining the name property of the camera targets that are produced by the vision job.



xx1300001098

Setup communication

The **Output to RAPID** functionality is displayed in the context window. The data from the camera is mapped to the `cameratarget` by choosing the desired data and arguments from the drop-down menus.

The data type `cameratarget` has a number of predefined components for data. These can be used in any way as long as the data type is considered. For a complete description of the data type `cameratarget` see *Technical reference manual - RAPID Instructions, Functions and Data types*.

View	Description
Part names	An explaining name of the part that the location- or inspection tool has identified (e.g. "Nut", "Screw", "Bolt", etc). This name will be transferred to the `name` argument of the resulting `cameratarget`.

Continues on next page

View	Description
Camera data mapping for part	• **Component**: The components corresponds to the arguments of the resulting `cameratarget`. • **Data type**: The data type that corresponds to the arguments of the resulting `cameratarget`. • **Group**: The group is the source of data, typically the configured inspection or location tools in the job. It is also possible to get data from the job itself, the input signals, the camera, or a constant. Each data group is an array that consists of many results (arguments). • **Result**: The results (arguments) of the group data. Typically x, y, z, and angle coordinates for the robot. A number of arguments are available depending on the data. These arguments are described in the online **Help** tab for each individual tool under the section *"... Tool Inputs/Outputs Properties"* • **Value**: The value of the Result. Typically a numerical value or a text string. The values displayed are transferred to the resulting RAPID `cameratarget`. **Note** The camera can only identify the rotation of the located part, z-angle. This value is converted to quaternion values in RAPID since the corresponding RAPID component is `pose`.

Mapping data

Use this procedure to make the vision data available to the RAPID program.

	Action
1	Click **Output to RAPID**.
2	Click **Add** in the **Item types** view to create a new item type.
3	Click **Rename** to give the item type an explaining name. For example the name of the part that the location- or inspection tool has identified (e.g. "Nut", "Screw", "Bolt", etc). The name is copied to the `type` argument of the resulting `cameratarget`.
4	Add vision data to the RAPID `cameratarget` by selecting them from the **Group** followed by the **Result** drop-down menus.
5	Check the result in the **Resulting camera target** view.
6	Save the job.
7	Run the RAPID instruction CamSetRunMode to update the controller on the current output to RAPID configuration.

For information about the different location tools and their parameters and settings, see the *Locate Part* section in the online **Help** tab.

For information about the different inspection tools and their parameters and settings, see the *Inspect Part* section in the online **Help** tab.

Note

It is not recommended to change the RobotStudio language after the configuration of an Integrated Vision system has started.

For more information, see *Changing the language on page 32*.

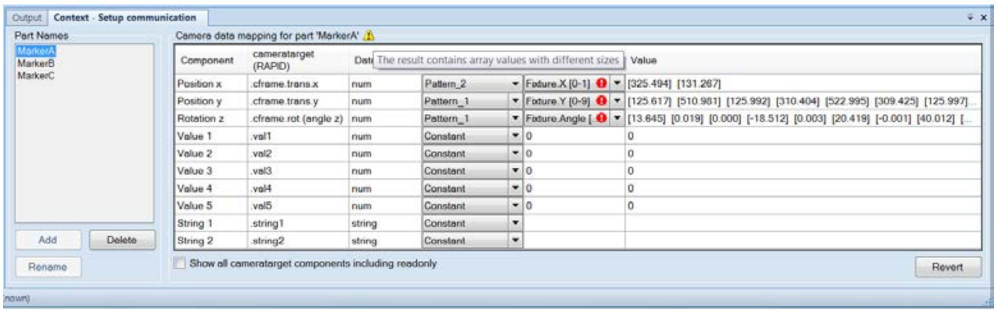
Continues on next page

Limitations

There are some restrictions to think about when completing the output configuration.

The number of values mapped to each component has to match. For example if a tool that finds 10 values, like **PatMax® Patterns (1-10)**, is mapped to the X-coordinate, it is not advisable to map tool output of another size to the other components. This would result in unpredictable behavior

Please observe warnings and cautions given by the Integrated Vision add-in:



xx1300001100

Furthermore, only values that are expected to change with each image acquisition are to be included in the result output. Other parameter values needed during runtime should preferably be fetched using the RAPID instruction `CamGetParameter`.

Most commonly used vision data

Location tool data

The following data results are typically used when locating parts:

Symbolic name	Description
Fixture.X	The x-coordinate of the located part.
Fixture.Y	The y-coordinate of the located part.
Fixture.Angle	The rotation of the selected part. **Note** The camera can only identify the rotation of the located part, z-angle. This value is converted to to quaternion values in RAPID since the corresponding RAPID component is `pose`.
Fixture.Score	A percentage value that references how well the found pattern resembles the trained pattern.
Pass	Set to 1 if the part is located, set to 0 if the part is not located.
Fail	Set to 0 if the part is located, set to 1 if the part is not located.

Inspection tool data

The following data results are typically used when inspecting parts:

Symbolic name	Description
Distance	The measured distance between two points.

Continues on next page

Symbolic name	Description
Result	Returns the string *Present* if the trained object is found, or *Not Present* if the trained object is not found.
String	Returns the value of an identified bar-code number as a string.

Job data

The following data results from the job are typically used:

Symbolic name	Description
Pass	Set to 1 if all vision tools that are set to **Include In Job Pass** are pass, set to 0 if one or more of the included vision tools are fail.
Fail	Set to 0 if all vision tools that are set to **Include In Job Pass** are pass, set to 1 if one or more of the included vision tools are fail.

5.9 I/O handling

Introduction

I/O is mostly used if the application includes external lighting that needs to be controlled by the camera. If there are no I/O signals connected to the camera, no configuration needs to be done.

Note

The **Inputs** and **Outputs** functions are only used to define the inputs and outputs of the camera, not the inputs and outputs of the robot controller.

Note

For more information see the *Cognex In-Sight®* manual, and the product manual and circuit diagram for the corresponding controller. See *References on page 7*.

General

The **Inputs** function is used to define the settings of the discrete inputs of installed *Cognex* camera I/O modules by customizing the name of the input line, setting the signal type, selecting the edge transition of the signal and forcing inputs to test the application.

The **Outputs** function is used to define the settings of the discrete outputs of installed *Cognex* camera I/O modules by customizing the name of the output line, setting the signal type, selecting tool results and forcing outputs to test the application. Some cameras have integrated outputs and LEDs. Those output signals are predefined and cannot be used for other purpose.

The settings are displayed in the **Context** window.

Configuring inputs

Use this procedure to configure the inputs of the vision camera, or any installed camera I/O modules.

	Action
1	Click **Inputs**. The available I/O signals are displayed in the **Context** window.
2	Use the **Signal Type**, **Edge Type**, and the **Force Input** drop-down menus to configure the desired behavior of the input.
3	To test the inputs, the camera must be in **Run Mode**.
4	Save the job.

For information about parameters and settings, see the *Inputs* section in the online Help tab.

Continues on next page

Configuring outputs

Use this procedure to configure the outputs of the vision camera, or any installed camera I/O modules.

	Action
1	Click **Outputs**. The available I/O signals are displayed in the **Context** window.
2	Use the **Signal Type**, **Job Result**, and the **Force Output** drop-down menus to configure the desired behavior of the output.
3	To test the outputs, the camera must be in **Run Mode**.
4	Save the job.

For information about parameters and settings, see the *Outputs* section in the online Help tab.

5.10 Preparing the RAPID program

5.10.1 RAPID snippets in RobotStudio

Introduction

Snippets are pieces of predefined RAPID code which can be inserted into the RAPID program.

A number of snippets have been created to facilitate the programming and commissioning of a vision system.

Tip

Use the `MoveToDetectedObject` snippet as a base when creating a new vision program.

Adding a snippet

The RAPID snippets are located in **Snippets** drop-down menu in the RobotStudio **RAPID** tab.

Use this procedure to add a snippet.

	Action
1	Request write access over the controller.
2	Expand the **RAPID** node of the controller browser and navigate to the **Program Modules.**
3	Create a new program module or open an existing module. Place the cursor on the desired insertion point for the snippet.
4	Click the **Snippets** drop-down menu, and select **Integrated Vision** to view the list of included snippets.
5	Click a snippet to include it in the program module.

For more information on how to program the controller using the RobotStudio **RAPID** tab, see *Operating manual - RobotStudio*.

5.10.2 Basic programming example

Introduction

This section describes how to write a basic vision guided robot program. The main purpose is to give an overview of what instructions need to be called. Some more advanced examples are also available.

For a detailed description of the vision specific RAPID instructions, functions, and data types, see *Technical reference manual - RAPID Instructions, Functions and Data types*.

Note

Before running a RAPID program, all previous configuration steps in this chapter has to be completed. See *Configuring Integrated Vision on page 39*.

Creating a basic RAPID program

The following example shows the basic steps to run a vision guided robot program. Error handling and other enhancements have been left out to provide a better overview.

The purpose of the following RAPID program is to move the robot to a position where the robot can pickup a part detected by the vision camera. It is based on the snippet `MoveToDetectedObject` found in RobotStudio. For more information, see *RAPID snippets in RobotStudio on page 66*.

```
1     ...
2     PERS wobjdata mywobj := ... ;
3     PERS tooldata mytool := ... ;
4     PERS robtarget myrobtarget := ... ;
5     CONST string myjob := "myjob.job";
6     VAR cameratarget mycameratarget;
7     ...
8     PROC MoveToDetectedObject()
9       CamSetProgramMode mycamera;
10      CamLoadJob mycamera, myjob;
11      CamSetRunMode mycamera;
12      CamReqImage mycamera;
13      CamGetResult mycamera, mycameratarget;
14      mywobj.oframe := mycameratarget.cframe;
15      MoveL myrobtarget, v100, fine, mytool \WObj:=mywobj;
16    ENDPROC
17    ...
```

Row	Comment
2 - 6	Declaration of data.
9 - 10	Set the camera to program mode and load the job.
11 - 12	Set the camera to run mode and acquire an image.
13	Get the vision result and store it in a camera target.

Continues on next page

Row	Comment
14	Copy the vision coordinates from the camera target to the object frame of the work object.
15	Move the robot to the pickup position.
-	(Pickup the part.)

For information on how to create a module, create a routine, and add RAPID-instructions, see chapter *Programming and testing* in *Operating manual - IRC5 with FlexPendant*.

Training the part grip location

Gripping a part is often not the same as moving the TCP to the target reported by the camera. Often this position must first be offset and rotated by some value to accommodate a good grip.

The easiest way to do this is by jogging the robot to the specified position and then modify the position.

	Action
1	Make sure that only one part is visible to the camera.
2	Run the program in the previous example down to the `MoveL` instruction and stop the execution. At this point the object frame of `mywobj` has been modified and the correct tool, `mytool`, is activated.
3	Verify that the part was successfully located by the vision camera.
4	Jog the robot to a good gripping position.
5	Mark the position `myrobtarget` and tap **Modify Position**.
6	Run the program from the top and make sure that the part is gripped according to the taught position.
7	Move the part and run the program from the top again.

5.10.3 Advanced programming examples

Introduction

The following examples show some of the more advanced features that can be used when running a vision guided robot program. The examples show the use of error handling, scene IDs, and other ways of increasing production and saving cycle time.

Data declaration

```
CONST string myjob := "myjob.job";
PERS robtarget myrobtarget :=
     [[100,200,300],[1,0,0,0],[0,0,0,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
VAR cameratarget mycameratarget;
...
PROC integratedvisionadvanced()
  VAR num mysceneid:=0;
  VAR num myexposuretime:=10;
  VAR bool targetsavailable := TRUE;
  ...
  ! Insert the code here
  ...
ENDPROC
```

Asynchronous loading

Loading a job asynchronously while moving the robot and then acquire an image as soon as the job finishes loading. This can optimize cycle time by having the robot performing other tasks while loading

```
CamSetProgramMode mycamera;
CamStartLoadJob mycamera, myjob;
!MoveJ myrobtarget, v100, fine, toolvision \WObj:=wobjvision;
CamWaitLoadJob mycamera;
CamSetRunMode mycamera;
CamReqImage mycamera;
```

Using the `SceneId` argument

Get an image and get the result as soon as it is available. Using the `SceneId` argument makes it possible to quickly notify the user if no target were found in the latest image.

```
CamReqImage mycamera \SceneId:=mysceneid;
CamGetResult mycamera, mycameratarget \SceneId:=mysceneid;
```

Note

The `SceneId` is generated as a unique serial number which is shared between all configured cameras using the RAPID instruction `CamReqImage`.

Changing parameters

Add 5 ms to the current exposure time

Continues on next page

```
myexposuretime := CamGetExposure (mycamera \ExposureTime);
myexposuretime := myexposuretime + 5;
CamSetExposure mycamera \ExposureTime:=myexposuretime;
```

Disable a vision tool

Disable a specific vision tool in the specified job. In order to avoid time consuming job loading, it may be desirable to configure various vision tools in a single job. By enabling/disabling the tools that are not currently used, the processing time can be minimized.

```
CamSetParameter mycamera, "Pattern_1.Tool_Enabled" \BoolVal:=FALSE;
```

Finding multiple parts

Find multiple parts in a single image and use the targets one by one. When there are no more targets left, the error `ERR_CAM_NO_MORE_DATA` is generated, see *ERROR handler on page 70*.

```
CamReqImage mycamera \SceneId:=mysceneid;
WHILE targetsavailable DO
  CamGetResult mycamera, mycameratarget \SceneId:=mysceneid;
  TPWrite "Current camera target is: "
       \Pos:=mycameratarget.cframe.trans;
ENDWHILE
```

Check if the job is already loaded

The job should only be loaded into the camera memory if it is not already loaded, in order to save cycle time.

```
IF CamGetLoadedJob(mycamera) <> myjob THEN
  CamLoadJob mycamera, myjob;
ENDIF
```

Sort parts by name

Sort parts depending on the part name property.

```
CamGetResult mycamera, mycameratarget;
IF mycameratarget.name = "wrench" THEN
  !Do something with the wrench
ELSEIF mycameratarget.name = "screwdriver" THEN
  !Do something with the screwdriver
ENDIF
```

ERROR handler

```
ERROR
  IF ERRNO = ERR_CAM_NO_MORE_DATA THEN
    TPWrite "There are no more targets originating from image with
         scene id "\Num:=mysceneid;
    targetsavailable:=FALSE;
    TRYNEXT;
  ENDIF
```

5.11 Starting production

Setting the camera to Run Mode

To start and run production the camera must be in run mode. The camera can manually be set to **Run Mode** from RobotStudio. This is mainly for testing purposes.

The recommended procedure is to set the run mode from RAPID using the instruction CamSetRunMode.

This page is intentionally left blank

6 Reference information

6.1 Relationships between coordinate systems

Introduction

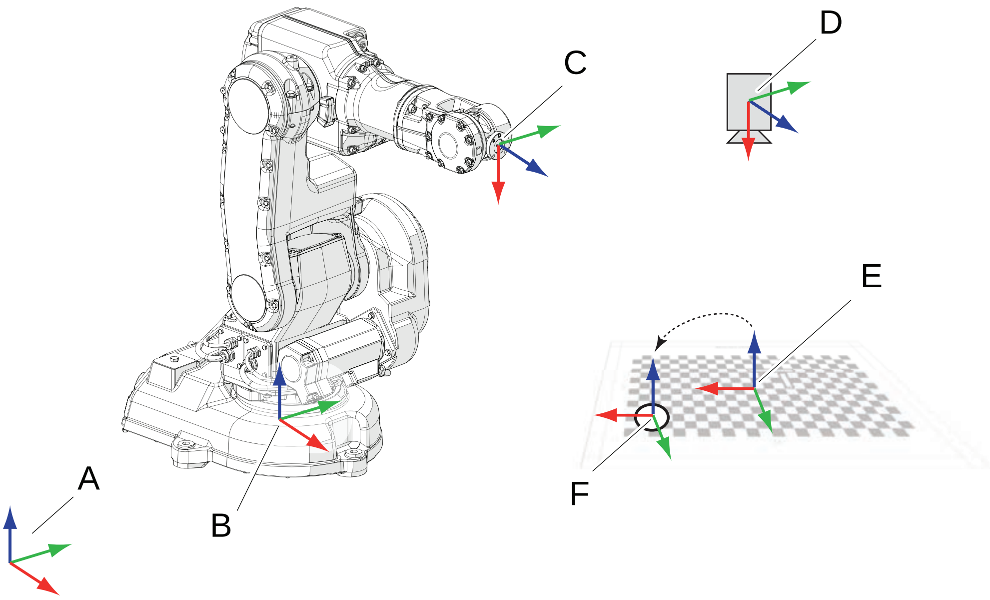
The robot controller has a number of built-in coordinate systems that all relate to each other in a predefined chain, WORLD, BASE, tool, work object, etc.

The vision camera also has coordinate systems to define the origin of the image and to define the distance in mm to the located parts within the image. Integrated vision provides means to synchronize the coordinate systems of the camera with the coordinate systems of the robot controller.

For more information about coordinate systems, and how to use them, see *Operating manual - IRC5 with FlexPendant* .

Coordinate systems in general

The image shows the most common coordinate systems of the robot controller. It is optional to define the WORLD coordinate system, but all other coordinate systems have to be measured by the robot controller.



xx1200000993

A	WORLD coordinate system.
B	BASE coordinate system.
C	Tool coordinate system (`tool0`).
D	Fixed camera position in space. The camera position is unknown to the robot controller unless the camera in held by the robot. **Note** If the camera is held by the robot, then the robot must go to the same position (`robtarget`) every time an image is acquired.

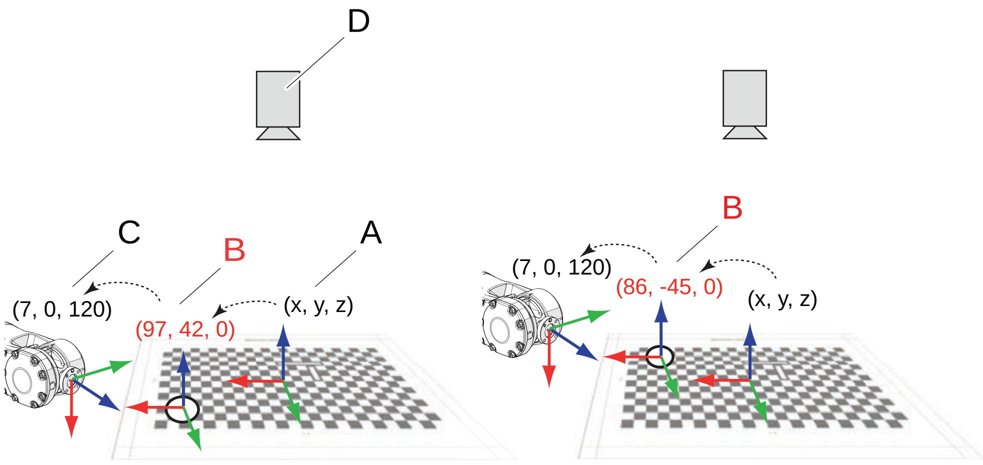
Continues on next page

E	*Work object - user frame* (`wobj.uframe`). Coincides with the camera frame when the camera to robot calibration has been performed.
F	*Work object - object frame* (`wobj.oframe`). It is recommended to use this coordinate system for the located part.

Calibrated camera frame (work object)

The following illustration and example shows the synchronization of the camera coordinate system with the coordinate systems of the robot controller.

> **Note**
>
> After the camera to robot calibration, the grid is no longer needed and can be removed.



xx1200000995

A	*Work object - user frame* (`wobj.uframe`). Coincides with the camera frame when the camera to robot calibration has been performed.
B	*Work object - object frame* (`wobj.oframe`). This coordinate system is used for the located part.
C	The gripping position of the robot (`robtarget`). (`tool0` is used in the illustration) The gripping position is related to the camera work object.
D	Fixed camera position in space (not held by the robot).

Example

The illustration to the left shows the basic setup when a part is located by the vision camera. The orientation, angle, has been left out to provide a better overview.

- The camera is located at a fixed position in space.
- The camera is calibrated by using a 10 mm calibration grid so that the camera can convert the image pixels to mm in x- and y-coordinates.

  The origin of coordinates is located at the intersection of the fiducial.
- The calibration grid is calibrated to the robot controller by the *work object - user frame*.

Continues on next page

The origin of the work object is also placed at the intersection of the fiducial.

- A part is located by the vision camera and the coordinates are sent to the robot controller.

  The grid spacing is 10 mm, giving the x-, y-, and z-coordinates (97, 42, 0).
- The coordinates of the part are written into the *work object - object frame*.
- The robot moves to pickup the part.

  The picking position has been modified in relation to the *work object - object frame*. For example 120 mm above the part and slightly offset in x by 7 mm, giving the coordinates (7, 0, 120).
- This means that the picking position can be the same no matter where the part is located.

In the illustration to the right, a new part has been located by the vision camera.

- The camera position is the same
- The calibration of the camera is the same
- The picking position is the same in relation to the part.
- The part has a new position, (86, -45, 0).

  The coordinates of the part are written into the *Work object - object frame*.
- The robot can now pickup the part.

6.2 Calibration theory

Introduction

The goal of most VGR applications is to provide positioning data from the camera to guide the robot. This requires the vision system to provide targets in a coordinate system shared with the robot. Creating a vision calibration can be divided into two steps, camera calibration and camera to robot calibration.

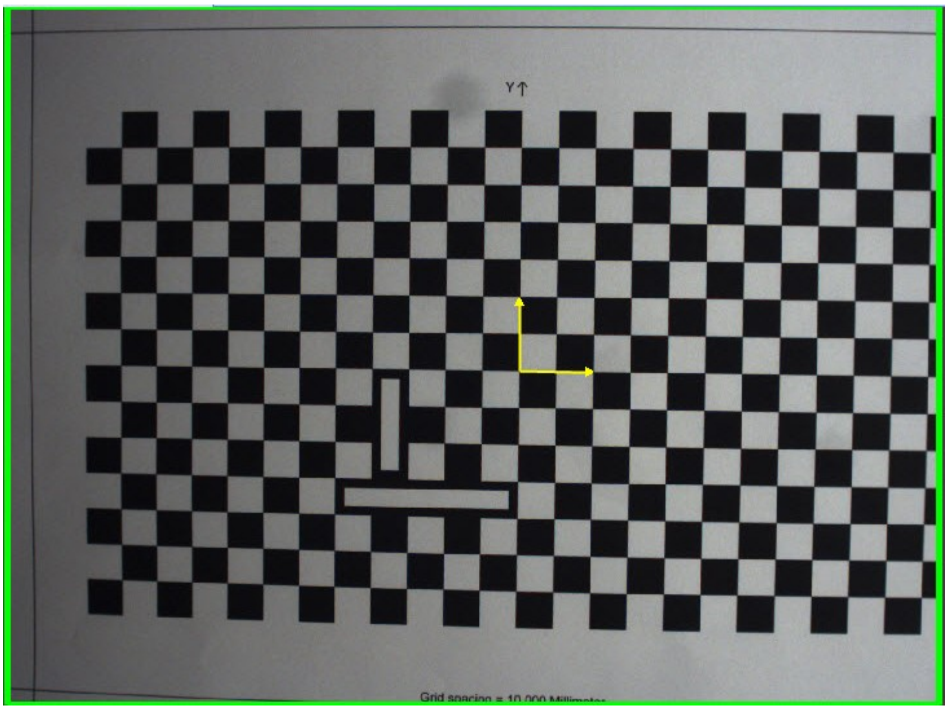
Camera calibration

The purpose of the camera calibration is to compute the transformation used for converting pixel coordinates in the image to physical coordinates in space (the relation between the camera frame and the checkerboard frame).

The Integrated Vision system works in 2D, meaning that all imaged parts must be located in a single calibrated plane or the result will not be accurate.

The Integrated Vision system provides several methods for calibrating the camera, but the most accurate and convenient method is to use a checkerboard plate. It is a pattern of known dimensions that also includes an encoded coordinate system. The relation between the camera and the checkerboard is fixed meaning that the camera always has to image the scene from the same point where it was calibrated.

In case the camera is mounted on the robot it has to move to the calibration pose before taking a photo during production. Once the camera has been calibrated any output from the image processing is expressed in millimeter in relation to the checkerboard origin.



xx1200000996

For information on how to perform a camera calibration, see *Camera calibration on page 54*.

Continues on next page

Camera calibration methods

The table below describes the available camera calibration methods in Integrated Vision.

The **X/Y Scale**, **Edge to Edge**, **X/Y Edge-to-Edge**, and **Circle** calibration options are useful in applications where the real-world measurements of the part or object are known.

Only the **Grid** calibration provides a reference mark that can be used for teaching a work object. The other calibration methods also create a pixel-to-mm relation, but they are less suitable for part location applications.

Note

The scale calibration options do not account for lens distortion, therefore, for the greatest degree of accuracy, make sure that the vision system is mounted perpendicular to the inspected surface. Otherwise, consider using **Grid** calibration to remove the distortion and more accurately calibrate the image.

Group	Description
X/Y Scale	Used when the X and Y scale values are known for calculating a simple calibration.
Edge to Edge	Used when distance between two edges are known for calculating a simple calibration.
X/Y Edge-to-Edge	Used when distance between two edge pairs are known. One in the horizontal direction, and another in the vertical direction. Combining the horizontal measurement and dimension with the vertical measurement and dimension creates two independent scales.
Circle	Used when the diameter of a circle is known.
Grid	The **Grid** calibration option creates a "map" of the image area by acquiring an image of a grid pattern of either dots or checkerboard squares. This option then relates the found features of the grid (the dots or intersections of squares) to the user-specified grid spacing of the grid pattern. This process creates a highly accurate computation that can then be used to transform distorted features into their correct shape.
Import	The **Import** calibration type is used to load a grid calibration file that has been exported during the configuration step. The calibration file (*.cxd*) must be located on the flash disk of the camera when loading the job that utilizes the calibration.

Camera to robot calibration

Once the camera has been calibrated any image processing output is expressed in the calibrated camera frame. To create a common frame for the robot and the camera, such that the robot can correctly interpret the position of the vision targets, the robot must be taught the calibrated camera frame.

The way to do this is to teach a work object that coincides with the calibrated camera frame. In other words, if a checkerboard pattern is used the work object is taught at the very same position (and orientation) as the checkerboard frame. A work object consists of two different frames – a user frame and an object frame. In this calibration the user frame is modified and the object frame is left untouched.

Continues on next page

For information on how to perform a camera to robot calibration, see *Camera to robot calibration on page 55*.

6.3 Best practise

6.3.1 Evaluate performance before adopting a solution

Introduction

When deciding to deploy a vision solution it is necessary to evaluate if the expected result can be achieved.

The safest way to ensure that required results can be achieved is to perform a test, and the closer closer the lab setup is to the intended installation the better the result.

It is of course not possible make a full setup, but it is important to consider critical factor such as:

- Required field of view
- Required accuracy
- Robustness

Identify the expectations

Before moving into a design phase it is necessary to identify the expectations so that appropriate actions can be acquired to maximize performance.

The concept of accuracy is relatively easy to understand, and in many cases there is a good perception of what is required of the system. However, the overall accuracy of a robot guided vision system, and even the vision system alone, is quite difficult to foresee.

Even though the individual vision tools such as a pattern recognition tool or caliper may have an extremely good theoretical accuracy, the true accuracy is highly dependent on external factors such as the product to be identified and the light setting.

Evaluate the vision accuracy

The general recommendation is to evaluate vision accuracy using the part or product that is actually going to be processed on the finished line. Furthermore, the light setup is critical to both accuracy and robustness.

Lighting is what many times makes or breaks final result, and important to note, something that may be difficult to change once the line has been designed and is being deployed.

In conclusion, when deploying vision systems it pays off to set up a vision trial in a lab environment.

Installation checklist

As good practice the following requirements shall be identified, quantified, and verified:

Requirement	Description
Samples	Collect good and bad samples of the actual customer product to be used for evaluation.

Continues on next page

Requirement	Description
Accuracy	What accuracy is required? The overall number combines robot accuracy, influence by part variation, lighting etc.
Tolerance	Can the part vary in size? Uniformly or irregularly?
Cycle time	The vision system requires processing time. Depending on the application this may or may not affect the cycle time.
Part positioning	Make sure to know the perspective from which the camera will observe the object. A simple thing like looking at the object from the side may affect the result.
Variations in the process	Apart from the verified variables, can something else change?
Lighting needs	Lighting is extremely important. Shield out ambient light and applying light that brings out the desired features of the part. Experimentation is the only reliable method.
Physical space constraints	Taking all factors into consideration such as field of view, lighting solution, and point of view. Does everything fit together?

6.3.2 How to mount the camera

Introduction

Depending on the application requirements and physical constraints, the camera may be mounted in different ways.

Generally it can be said that mounting the camera on a fixed structure is more efficient unless requirements are such that the camera must be carried by the robot.

Stationary camera

A stationary camera generally provides faster cycle times since the robot does not have to stop on its path to acquire an image. Setup and calibration is generally easier with fixed cameras since the point from which the image is acquired is fixed.

When mounting the camera on a fixed structure it is important that the camera is not subject to vibrations which can cause motion blur.

Robot held camera

If it is not possible to mount the camera at a fixed location, the camera can also be mounted on a moveable part of the manipulator. In that case the camera would generally be mounted on the robot tool to avoid occlusion. Each application is different, and tool designs as well as dress packs for the cables differ from case to case.

When placing a camera on a moving position it is the user's responsibility to assure that the camera is not subjected to mechanical forces greater than what is specified in the camera specification. The cables are of a flexible type, but wear depends greatly on both the cable routing and the programmed robot path.

CAUTION

When using a robot held, or by other means moving camera, it is important to have a good cable routing.

When routing the cables caution has to be taken to avoid mechanical stress on the connectors, assure sufficient bend radius for the cables, and minimize the wear on the cables. It is also recommended to fit the cables with extra wear protection at the attachment points and at especially exposed areas.

6.3.3 Obtain accuracy

Introduction

This section contains useful tips to optimize the overall accuracy of a vision guided robot solution. Knowing which factors influence accuracy can greatly help avoiding the most common pitfalls.

Training the vision tool

When training a vision system to recognize or measure a part, first choose which section of the part that is of interest. This often turns into a trade-off between what relevant features are available and how consistent they are.

Consistency of the feature to be detected is critical and comes into play in various different ways. Is the relevant feature present and similar looking on different individual parts, and can it still be recognized if the part is moved across the field view? This is where lighting comes into play. If the appropriate lighting is not applied, the image processing will be prone to fail either when the part is presented to the camera at different positions, or because the feature to be detected is not sufficiently visible on all parts.

To summarize, it is important to decide which section of the part to measure, and to select illumination that brings out the interesting features of that section in a consistent way.

For more information, see *Obtain good lighting on page 84*.

The field of view

Many times it may not be necessary to fit the whole part within the field of view in order to perform the desired measurement or identification.

Most important is that relevant features are clearly and consistently visible. Reducing the field not only increases the pixel-to-mm ratio which provides higher accuracy, but it also reduces the area where appropriate lighting needs to be applied. Furthermore it should be said that the perspective from which the camera views the part can influence the result.

A general rule of thumb is that the camera shall, if possible, image the part from right angle. In cases where this is not possible a non-linear calibration such as the checkerboard helps compensate for non perpendicular viewing angles. While having the capability to accurately translate coordinates seen from a tilted angle, it does not automatically solve the problem that vision tools may fail to operate robustly when the image is distorted due to perspective or lens effects.

As a remedy in cases when the image is distorted either by the optics or the viewing angle it is possible to apply a rectifying filter that unwraps the image. Under **Add Part Inspection Tool** and **Image Filter Tools** select **Transform**. This tool takes the result from a grid calibration and uses that information to calculate an "undistorted" version of the image. For subsequently added vision tools the user may choose whether they shall run on the rectified image or the original image.

Continues on next page

The checkerboard / calibration plane

As already mentioned there are various essential factors to the imaging of a scene, and the subsequent image processing that greatly influence the accuracy of the result.

But none of that matters if the geometry, or calibration, of the system is not true to the physical setup. The camera coordinate system is, normally, established by placing a calibration plate with an origin marker under the camera and performing a calibration routine.

The calibration of the camera is most commonly referenced by the robot in the form of a work object for which the user frame is placed in the exact same position as the camera coordinate system.

It is clear that if the work object is not accurately defined to the position of the origin marker of the camera accuracy is lost. But what is more, the results reported by the vision system are always the projection of identified and measured features as projected on the calibration plane. This means that in order to obtain accurate results the features to be measured shall be located in the calibration plane.

6.3.4 Obtain good lighting

Introduction

It is important to understand that good lighting for machine vision is not the same as good lighting perceived by a human being.

The specific features of the part to be inspected shall be clearly and consistently illuminated so that the result of the image processing is repeatable throughout the full active field of view.

Common problems

The most common problems arise from uneven light distribution, reflections, shadows, and glare. It is critical to the outcome of the image processing that such unwanted artifacts are eliminated or at least minimized. Shiny parts such as metallic items of transparent plastic surfaces generally require careful consideration before selecting a lighting system.

There is no single universal lighting technique that is suitable for all parts and situations. However, by understanding the fundamentals of how image processing works and how lighting can be adjusted to provide good input data, performance and robustness can be greatly improved.

Lighting techniques

As a rule of thumb it can be said that the key to control the light, is to shield out unwanted light and to apply light suitable for the application. To rely on ambient light (partially or fully) is not advisable.

One commonly used technique to shield out light is to use a light hub that provides an enclosure around the work piece to be observed. The inside of the box is fitted with the type of light that best brings out the interesting features. Another method for blocking out unwanted light is to use a combination of an optical bandpass filter mounted on the lens while applying color lighting.

The filter is matched so as only to let through light of the same wavelength (or color) as that applied by the lighting system.

Another technique that is used very successfully in many situations is called backlighting. Instead of illuminating the face of the part, light is instead applied from behind the product so that the contours are brought out. This provides a simplified black on white image that is often easier for the vision tools to process.

How to select a lighting system

The safest way to determine what is an appropriate lighting system for the application at hand is, by far, to perform a practical vision evaluation using the actual part and applying different light sources from different angles.

During an evaluation it is important to consider factors such as stand-off distance, the size of the field of view, what ambient light exists and which vision tool is to be used for the task. All of these factors affect one another and must be tuned to match.

6.3.5 Structuring the vision job

Introduction

Often a vision system is required to detect different parts, either in each image or between different production shifts.

Different approaches can be applied to structure the vision job depending on the current requirements:

Create separate jobs for each product or task

Creating separate jobs provides a structured setup but jobs take several seconds, even up to a minute to load. Each job also contains its own camera calibration, which may be useful if the different tasks require different calibrations.

Create a single job for all products and tasks

Create a single job that contains vision tools for all different parts and production scenarios.

This approach is typically used if the system is required to look for different parts in a single cycle. Adding tools makes the job file bigger, and finally the maximum limit is reached. Unless tools are disabled they all execute with each image acquisition which slows down the vision execution considerably. When placing all vision tools in a single job the various tools are typically switched on and of using RAPID calls, see *Enabling and disabling vision tools during runtime on page 87*.

6.3.6 Init routine

Description

Always run an initiation routine after powering up or restarting the controller. This ensures that the proper job is loaded, and that the controller and the cameras are in the correct mode.

Event	Description
After a power failure:	• The camera will loose the job. • The controller will loose the output to RAPID configuration.
After a controller restart:	• The controller will loose the current status of the camera, program mode/run mode. • The controller will loose the output to RAPID configuration.

Example

```
PROC IV_Init(VAR cameradev cam,string jobname)
  VAR num maxloadtime:=15;

  CamSetProgramMode cam;
  CamLoadJob cam,jobname\MaxTime:=maxloadtime;
  CamSetRunMode cam;

  MoveAbsJ safepos,v100,fine,tool0;

ERROR
  IF ERRNO=ERR_CAM_BUSY THEN
    TPWrite "ERR_CAM_BUSY. Calling RETRY";
    WaitTime 1.0;
    RETRY;
  ELSEIF ERRNO=ERR_CAM_MAXTIME THEN
    TPWrite "ERR_CAM_MAXTIME. Increasing timeout by 10s and calling
         RETRY";
    maxloadtime:=maxloadtime+10;
    WaitTime 1.0;
    RETRY;
  ELSEIF ERRNO=ERR_CAM_NO_PROGMODE THEN
    TPWrite "ERR_CAM_NO_PROGMODE. Setting camera to program mode
         and calling RETRY";
    WaitTime 1.0;
    CamSetProgramMode cam;
    RETRY;
  ENDIF
ENDPROC
```

6.3.7 Enabling and disabling vision tools during runtime

Description

Sometimes it is known which type of part to look for in the image. If so, the vision tools that are not currently needed can be disabled to reduce the camera processing time.

Consider an example where two vision tools, `Pattern_1` and `Pattern_2` which produce items of type `Item1` and `Item2` respectively. The following procedure describes how to turn the two vision tools on and off.

The disabled tool still produces results with the values from the latest active execution. In order to not use these targets, sort them out in the RAPID program.

Example

This procedure shows how to turn tools on and off to shorten the camera processing time. It also shows how to handle the result queue, when disabling tools.

```
PROC Enable_Disable_Tools(VAR cameradev cam,bool enabletool1,bool
    enabletool2)
  VAR cameratarget mycameratarget;
  VAR string tool1_propertyname:="Pattern_1.Tool_Enabled";
  VAR string tool2_propertyname:="Pattern_2.Tool_Enabled";
  VAR string tool1_corresp_item_name:="Item1";
  VAR string tool2_corresp_item_name:="Item2";
  VAR num maxresulttime:=5;
  CamSetProgramMode cam;
  CamSetParameter cam,tool1_propertyname\BoolVal:=enabletool1;
  CamSetParameter cam,tool2_propertyname\BoolVal:=enabletool2;
  CamSetRunMode cam;
  CamReqImage cam;
  CamGetResult cam,mycameratarget;
  !Before using the result, make sure that it originates from an
      enabled tool.
  !Disabling tools reduces the processing time, but the latest
      result produced by the tool is still communicated.
  IF mycameratarget.name=tool1_corresp_item_name AND enabletool1
      THEN
    TPWrite "Received target of type "+mycameratarget.name+" with
        position "\Pos:=mycameratarget.cframe.trans;
    !Run the robot..
  ELSEIF mycameratarget.name=tool2_corresp_item_name AND enabletool2
      THEN
    TPWrite "Received target of type "+mycameratarget.name+" with
        position "\Pos:=mycameratarget.cframe.trans;
    !Run the robot..
  ENDIF
  !Flush the result queue to get rid of any remaining targets.
  CamFlush cam;
ERROR
  IF ERRNO=ERR_CAM_BUSY THEN
    TPWrite "ERR_CAM_BUSY. Calling RETRY";
```

Continues on next page

```
    WaitTime 1.0;
    RETRY;
  ELSEIF ERRNO=ERR_CAM_MAXTIME THEN
    TPWrite "ERR_CAM_MAXTIME. Increasing timeout by 5s and acquiring
         new image";
    CamFlush cam;
    CamReqImage cam;
    maxresulttime:=maxresulttime+5;
    WaitTime 1.0;
    RETRY;
  ELSEIF ERRNO=ERR_CAM_NO_RUNMODE THEN
    TPWrite "ERR_CAM_NO_RUNMODE. Setting camera to run mode and
         calling RETRY";
    WaitTime 1.0;
    CamSetRunMode cam;
    RETRY;
  ELSEIF ERRNO=ERR_CAM_NO_PROGMODE THEN
    TPWrite "ERR_CAM_NO_PROGMODE. Setting camera to program mode
         and calling RETRY";
    WaitTime 1.0;
    CamSetProgramMode cam;
    RETRY;
  ENDIF
ENDPROC
```

6.3.8 Avoid running out of space on the camera

Description

The flash disc of the camera has got a limited storage space. If a robot cell requires a lot of different jobs, the camera will at some point not be able to store all jobs.

It can then be convenient to keep only the necessary jobs on the camera and store the rest on the robot controller. When they are needed they can be uploaded from the controller flash disk to the camera.

Moving files between the camera and the controller can be carried out either using RAPID or the FlexPendant explorer, where the flash disks of the cameras appear next to the controller drive.

Below is an example showing how to move files from the home catalog of the system to the camera.

See also *Backup a camera to the controller on page 90*.

Example

```
VAR string campath;
VAR string controllerpath;
...
campath := CamGetName(mycamera) + ":/" + "myjob.job";
controllerpath := "HOME:/myjob.job";
CopyFile controllerpath, campath;
```

6.3.9 Backup a camera to the controller

Description

The contents of a camera are not automatically backed up when taking a regular backup of the robot controller. However, the following RAPID routine copies all of the files from a camera to the home directory of the controller. This can be used as a service routine that can be called before taking the regular backup.

Tip

The code is available as a snippet in RobotStudio. There is also a snippet for restoring the files.

Example

```
PROC BackupCamToCtrl(var cameradev cam,bool replaceexistingfiles)
  VAR string ctrldirname:="HOME/IV/";
  VAR dir camdirectory;
  VAR string camdirname;
  VAR string tempfilename;
  VAR string tempcamfilepath;
  VAR string tempctrlfilepath;
  ...
  camdirname:=CamGetName(cam)+":/";
  ctrldirname:=ctrldirname+CamGetName(cam)+"/";
  MakeDir ctrldirname;
  OpenDir camdirectory,camdirname;
  WHILE ReadDir(camdirectory,tempFileName) DO
    tempcamfilepath:=camdirname+tempfilename;
    tempctrlfilepath:=ctrldirname+tempfilename;
    CopyFile tempcamfilepath,tempctrlfilepath;
  ENDWHILE
  CloseDir camdirectory;
ERROR
  IF ERRNO=ERR_FILEEXIST THEN
    IF replaceexistingfiles THEN
      RemoveFile tempctrlfilepath;
      RETRY;
    ELSE
      TRYNEXT;
    ENDIF
  ENDIF
ENDPROC
```

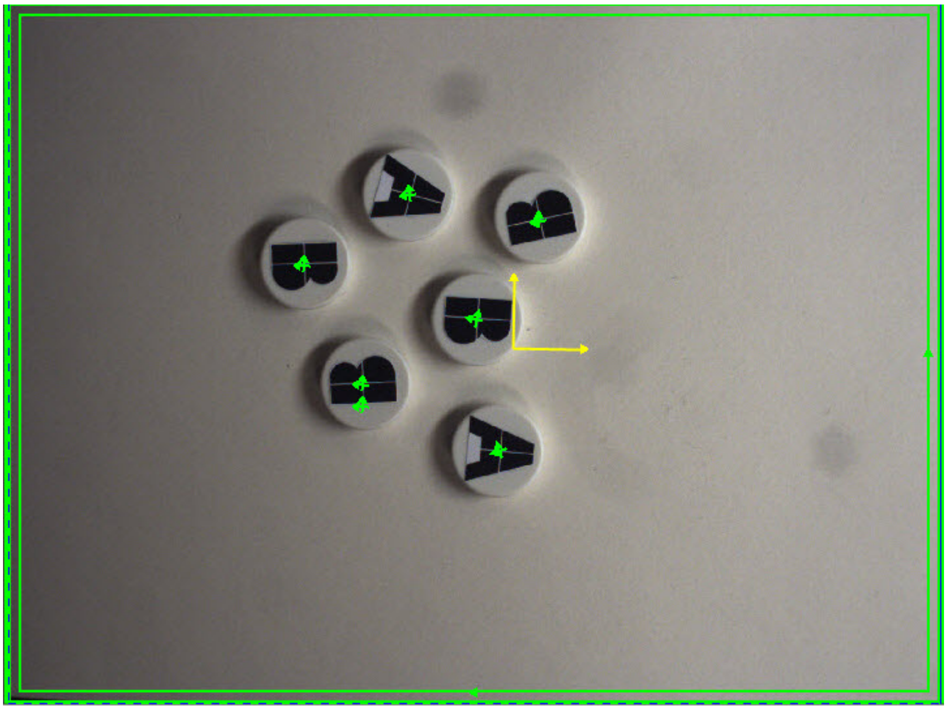
6.3.10 Sort items of different types

Description

Sometimes it is needed to distinguish between products of different types. The following example describes how to distinguish between two chips which have the letters A respective B printed on them.

The solution can be divided into three basic steps:

1. Train a vision job that is able to distinguish between the two chip types
2. Configure the output to RAPID so that the RAPID program can retrieve both the position and the type of the object.
3. Program a RAPID program that retrieves vision targets and checks which type it is before acting on it.



xx1300000190

Train the vision job

First, the vision system must be taught to distinguish between the two part types.

This problem can be solved in various ways using the available vision tools. In this case two different part location tools, *PatMax®*, are trained on the features that differ between the two product types - the letters A and B.

There are two reasons for not including the circle in either of the models. The first reason is that the circle is identical for both parts, making them more similar from a vision perspective. The second reason is that the circular feature makes up a large percentage of the model content while not providing any information about the angle of the part. This results in angular uncertainty.

Continues on next page

Configure the output to RAPID

Second, the output to RAPID must be configured so that the information about which part is which is preserved and can be used for the robot to act on. To do this, two items are configured and named after the part types they represent – A_Chip and B_chip. The outputs of the tool that identifies the A-chip is connected to the components of A_Chip and the same is done for the B-chip.

Program the RAPID program

Finally, the RAPID program must be programmed to act based on the type of the identified target.

Example

```
...
CamGetResult mycamera, mycamtarget;
IF mycamtarget.name = "A_Chip" THEN
  !Do something with the A-Chip
ELSEIF mycamtarget.name = "B_Chip" THEN
  !Do something with the B-Chip
ENDIF
...
```

6.3.11 Finding multiple items of the same type

Description

Often the vision task involves finding multiple parts of the same type spread across the field of view of the camera. Configuring such a task requires some configuration of the vision job and the RAPID program to be made.

1. Typically a vision tool that finds multiple items is used, `PatMax[1-10]` or `Blob[1-10]`.
2. In the **Output to RAPID** dialog the vision tool is linked to a single item type, for example called *Part_A*. This means that for each found item a camera target with the name property *Part_A* will be produced and sent to the vision queue.
3. The task of the RAPID program is to loop through each produced `cameratarget` and pick up the corresponding object.

Example

The following example acquires an image and moves to each of the reported targets. The camera job may produce multiple targets for each acquired image.

```
PROC Multiple_Target_Pick(VAR cameradev cam)
  VAR bool continueloop:=TRUE;
  VAR num maxresulttime:=5;
  VAR cameratarget mycameratarget;
  VAR num zoffset:=200;
  CamReqImage cam;
  WHILE continueloop DO
    CamGetResult cam, mycameratarget;
    camwobj.oframe:=mycameratarget.cframe;
    MoveL offs(picktarget,0,0,zoffset), v500, z0, picktool
         \WObj:=camwobj;
    MoveL picktarget, v100, fine, picktool \WObj:=camwobj;
    WaitTime 1.0;
    MoveL offs(picktarget,0,0,zoffset), v500, z0,picktool
         \WObj:=camwobj;
    IF CamNumberOfResults(cam)<1 THEN
      continueloop:=FALSE;
    ENDIF
  ENDWHILE
ERROR
  IF ERRNO=ERR_CAM_BUSY THEN
    TPWrite "ERR_CAM_BUSY. Calling RETRY";
    WaitTime 1.0;
    RETRY;
  ELSEIF ERRNO=ERR_CAM_MAXTIME THEN
    TPWrite "ERR_CAM_MAXTIME. Increasing timeout by 5s and acquiring
         new image";
    CamFlush cam;
    CamReqImage cam;
    maxresulttime:=maxresulttime+5;
```

Continues on next page

```
      WaitTime 1.0;
      RETRY;
    ELSEIF ERRNO=ERR_CAM_NO_RUNMODE THEN
      TPWrite "ERR_CAM_NO_RUNMODE. Setting camera to run mode and
           calling RETRY";
      WaitTime 1.0;
      CamSetRunMode cam;
      RETRY;
    ENDIF
  ENDPROC
```

6.3.12 Always check that the vision target is within expected limits

Description

When using sensors it is important to always make sure that a detected position is within the expected work space before attempting approach the position with the robot.

A poorly trained model or calibration failure could produce targets in unexpected positions. As an example the following procedure, also available as a snippet, should be used to minimize the risk of such problems.

Example

```
IF (CamCheckLimits(mycameratarget.cframe, -100, 100, -100, 100,
     -90, 90)) THEN
  !Perform move instruction
ELSE
  !Perform recovery routine
ENDIF
FUNC BOOL CamCheckLimits(pose current_pose, num X_min, num X_max,
     num Y_min, num Y_max, num Angle_min, num Angle_max)
  !Checks that the pose is within the specified limits.
  IF (current_pose.trans.X < X_min) RETURN FALSE;
  IF (current_pose.trans.X > X_max) RETURN FALSE;
  IF (current_pose.trans.Y < Y_min) RETURN FALSE;
  IF (current_pose.trans.Y > Y_max) RETURN FALSE;
  IF (EulerZYX(\Z, current_pose.rot) < Angle_min) RETURN FALSE;
  IF (EulerZYX(\Z, current_pose.rot) > Angle_max) RETURN FALSE;
  RETURN TRUE;
ENDFUNC
```

This page is intentionally left blank

7 RAPID components

About the RAPID components

This is an overview of all instructions, functions, and data types in *Integrated Vision*.

For more information, see *Technical reference manual - RAPID Instructions, Functions and Data types*.

Instructions

Instructions	Description
CamFlush	`CamFlush` is used to flush (remove) the `cameratarget` collection for the camera.
CamGetParameter	`CamGetParameter` is used to get named parameters that the camera may expose. The user has to know the name of the parameter and its return type in order to retrieve its value.
CamGetResult	`CamGetResult` *(Camera Get Result)*) is used to get a camera target from the vision result collection.
CamLoadJob	`CamLoadJob` *(Camera Load Job)* loads a camera task, *job*, describing exposure parameters, calibration, and what vision tools to apply.
CamReqImage	`CamReqImage` *(Camera Request Image)* orders the camera to acquire an image.
CamSetExposure	`CamSetExposure` (*Camera Set Exposure*) sets camera specific data and makes it possible to adapt image parameters depending on ambient lighting conditions.
CamSetParameter	`CamSetParameter` is used to set different named camera parameters that a camera may expose. With this instruction it is possible to change different parameters in the camera in runtime. The user has to know the name of the parameter and its type in order to set its value. User created parameters can only be of type `EditString`, `EditInt`, or `EditFloat`.
CamSetProgramMode	`CamSetProgramMode` *(Camera Set Program Mode)* orders the camera to go to program mode (offline).
CamSetRunMode	`CamSetRunMode` *(Camera Set Running Mode)* orders the camera to go to run mode (online), and updates the controller on the current output to RAPID configuration.
CamStartLoadJob	`CamStartLoadJob` will start the loading of a job into a camera, and then the execution will continue on the next instruction. When loading is in progress other instructions can be executed in parallel.
CamStartSetParameter	`CamStartSetParameter` is used to start the set-operation of a parameter setting in the camera. The execution of the RAPID program will continue with next instruction/function. The camera will be busy running the set-parameter operation and will not perform any other request before this operation is completed with `CamWaitSetParameter`. The controller can have 10 pending `CamStartSetParameter` operations. When the controller has 10 pending requests, all the requests has to be confirmed with `CamWaitSetParameter` before any new `CamStartSetParameter` is ordered. User created parameters can only be of type `EditString`, `EditInt`, or `EditFloat`.

Continues on next page

Instructions	Description
CamWaitLoadJob	`CamWaitLoadJob` *(Camera Wait Load Job)* will wait until the loading of a job into a camera is ready.
CamWaitSetParameter	`CamWaitSetParameter` will wait until the set-operation of the parameter is done in the camera and return with status.

Functions

Functions	Description
CamGetExposure	`CamGetExposure (Camera Get Exposure)` is a function that reads the current settings for a camera. With this function and with the instruction `CamSetExposure` it is possible to adapt the camera images depending on environment in runtime.
CamGetLoadedJob	`CamGetLoadedJob` *(Camera Get Loaded Job)* is a function that reads the name of the current loaded job from the camera and returns it in a string.
CamGetMode	`CamGetMode` (*Camera Get Mode*) is a function that is used to get the mode of the camera. The function returns the mode in the data type `camerastatus`.
CamGetName	`CamGetName (Camera Get Name)` is used to get the configured name of the camera.
CamNumberOfResults	`CamNumberOfResults (Camera Number of Results)` is a function that reads the number of available vision results and returns it as a numerical value.

Data types

Data types	Description
cameradev	`cameradev` *(camera device)* is used to define the different camera devices which can be controlled and accessed from the RAPID program. The data type `cameradev` is used for instructions and functions communicating with a camera. The names of the camera devices are defined in the system parameters and, consequently, must not be defined in the program.
camerastatus	`camerastatus` (*camera status*) is used to define the different camera modes accessed by `CamGetMode`. Possible values for `camerastatus` are `CAMERA_DISCON`, `CAMERA_STANDBY`, and `CAMERA_RUNNING`.
cameratarget	`cameratarget` is used to exchange vision data from the camera image to the RAPID program.

Note

The `CamSetExposure` instruction depends on the trigger type. When the trigger type is set to `Camera`, the image acquisition parameters of the camera, such as, *exposure time*, are loaded after the camera captures the subsequent image. For all other trigger parameter modes (for example, *External*), the parameter values are loaded soon after the camera receives the signal to acquire the image, hence the subsequent image reflects the expected modifications.

Index

ABB AB
Robotics & Discrete Automation
S-721 68 VÄSTERÅS, Sweden
Telephone +46 10-732 50 00

ABB AS
Robotics & Discrete Automation
Nordlysvegen 7, N-4340 BRYNE, Norway
Box 265, N-4349 BRYNE, Norway
Telephone: +47 22 87 2000

ABB Engineering (Shanghai) Ltd.
Robotics & Discrete Automation
No. 4528 Kangxin Highway
PuDong New District
SHANGHAI 201315, China
Telephone: +86 21 6105 6666

ABB Inc.
Robotics & Discrete Automation
1250 Brown Road
Auburn Hills, MI 48326
USA
Telephone: +1 248 391 9000

abb.com/robotics

3HAC044251-001, Rev R, en