Asyril
EYE+ Provider

Version 1.0.0

User's guide

July 13, 2022

[Revision history]

Version	Date	Contents
1.0.0	13.07.2022	First version.

Contents

1. Introduction

This document is a user's guide of the EYE+ provider that is the CAO provider for the EYE+ Smart Control System manufactured by Asyril.
The EYE+ provider connects with the EYE+ controller with ethernet **TCP/IP messaging only**, managing all low-level communication.

2. Outline of provider

2.1. Outline

This provider gives a straightforward access to the base control commands of an EYE+ controller.
All EYE+ commands can be addressed through CaoController::Execute() call.

To configure EYE+ and tune recipes use EYE+ studio interface.

Table 2-1 Asyril EYE+ provider

File name	CaoProvAsyrilEYEPlus.dll
ProgID	CaoProv.Asyril.Eyeplus
Registry registration	regsvr32 CaoProvAsyrilEYEPlus.dll
Registry un-registration	regsvr32 /u CaoProvAsyrilEYEPlus.dll

2.2. Method and property

2.2.1. CaoWorkspace::AddController method

Syntax

AddController(<CtrlName>, <ProvName>, <ExecMachineName>, <OptionStr>)

CtrlName : [in] Controller name, Arbitrary string (Ex: "EYEPLUS")

ProvName : [in] Provider name. (Fixed to "CaoProv.Asyril.Eyeplus")

ExecMachineName : [in] Execution machine name of provider. Not used. Leave it empty.

OptionStr : [in] Option character string

Option	Meaning
Conn=<connection parameter>	Set the communication parameter. The only valid communication protocol is TCP.
Timeout=<Delay>	Set the TCP socket communication timeout. Unit in seconds. A timeout message error will be sent as Execute output If a send or receive takes too much time to end. If not specified, timeout = 33s.

Examples

Eyeplus = cao.AddController("EYEPLUS", "CaoProv.Asyril.Eyeplus", "",
"conn=TCP:192.168.0.50:7171,Timeout=30")

3. Command reference

3.1. CaoController::Execute(« <Command name> ») command

Syntax

Response = Eyeplus.Execute("<EyeCommand>")

EyeCommand : [in] EYE+ command to send. See EYE+ user guide for command list.

Response : [out] EYE+ response (string).

EYE+ command can be easily sent through the Execute command.

Note that the provider will never introduce controller error. If communication error occurred for some reason (disconnection, not connected, timeout while sending or receiving, timeout while trying to reconnect), an error message will be passed in "Response".

The list of command that can be sent to EYE+ is described here.

3.1.1. List of control commands

#	<Command>	Description
1	"start production <recipe_id>"	This command must be called to start EYE+ in production state using the right recipe. • <recipe_id> - the recipe's unique identifier. The parameter must be an integer between 1 and 65535. Return "200" on success and error code on failure.
2	"stop <state>"	This command is used to stop an EYE+ state. Return "200" on success and error code on failure.
3	"get_part"	This command is used to request one or more parts from EYE+. This is a blocking command, meaning it will keep going until it gets a response. Return "200 x=<x> y=<y> rz=<rz>", where <x>, <y>, <rz> are coordinates and orientation of the part, else refer to error codes.
4	"prepare_part"	Prepare one or more parts from EYE+, prepared parts can be retrieved later using the command get_part. Return "200" on success and error code on failure.

<table>
<tr><td>5</td><td>“get_parameter <parameter>”</td><td>Returns the value of the parameter specified as the command argument. Valid parameters are described in EYE+ documentation.

Return “200” on success and error code on failure.</td></tr>
<tr><td>6</td><td>“set_parameter <parameter> <value>”</td><td>Assign a value to the parameter specified as the command argument. Valid parameters are described in EYE+ documentation.

Return “200 <response>”, returns 200 followed by value of the parameter specified if command succeeded, else refer to error codes.</td></tr>
<tr><td>7</td><td>“can_take_image <boolean>”</td><td>Tells EYE+ that it can acquire an image. This command should be called when the field of view is clear of any object that could obstruct the image acquisition.

<table><tr><td>#</td><td><boolean></td><td>description</td></tr><tr><td>1</td><td>true</td><td>Disable the image acquisition.</td></tr><tr><td>2</td><td>false</td><td>Enable the image acquisition.</td></tr></table>
Return “200” on success and error code on failure.</td></tr>
<tr><td>8</td><td>“force_take_image”</td><td>The command force_take_image forces EYE+ to acquire an image as soon as possible.

Return “200” on success and error code on failure.</td></tr>
<tr><td>9</td><td>“feeder <command>”</td><td>Send a raw Asycube command from EYE+ to the connected Asycube.

Return “200 <response>”, returns 200 followed by the response from the Asycube if the command succeeded, else refer to error codes.</td></tr>
</table>

3.1.2. Error messages

The following table details the error messages defined by the provider. These errors will not interrupt the program execution since it is only the Execute command output.

Syntax

<Error code> <Error message>

The error message may start with error code 4xx or 5xx. In this case, it is a specific EYE+ error. You can find the list of EYE+ error codes here.

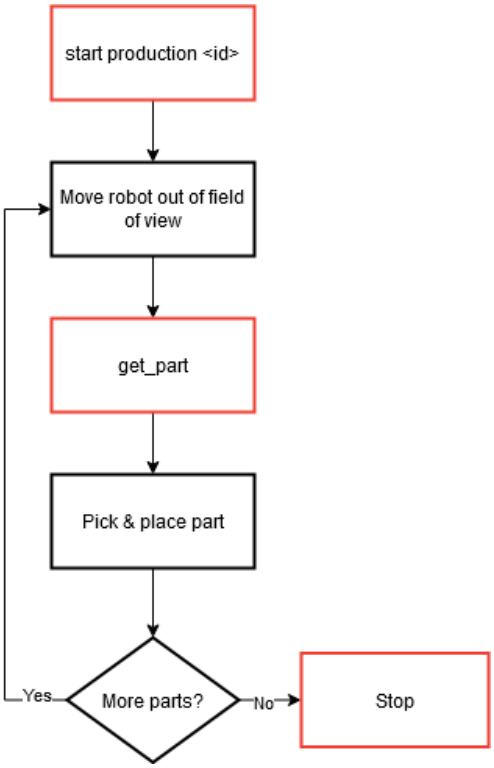
If the error code starts with 6xx, it is a specific plugin error. The table below describes all possible plugin errors you may get.

Error code	Error message	Context	Typical Cause
601	Communication error	On any "Execute" call.	Internal socket communication error.
602	Client configuration error	Wrong connection parameter provided in cao.AddController().	Connection parameters provided does not fit with IP address and/or port number format.
603	Connection failed	Wrong connection parameter provided in cao.AddController().	Connection parameters provided are wrong or cable not connected.
604	Timeout occurred	On any "Execute" call.	The command sent or response reception takes too much time (more that timeout value).
699	Generic error	On any "Execute" call.	Unknown error occurred.

4. Sample program

This example shows a Denso PacScript that illustrate a EYE+ pick and place scenario.

It is necessary at this stage to have already performed the camera configuration, as well as the creation of a recipe with its hand-eye calibration.



```
'!TITLE "Example of EYE+ provider direct use"

Sub Main
	' Start Communication by creating the EYE+ provider
	Dim Eyeplus as Object
	Eyeplus = cao.AddController("eyeplus", "CaoProv.Asyril.Eyeplus","","conn=TCP:192.168.0.50:7171,Timeout=30")

	Set IO[24] ' Open Gripper

	'Start EYE+ in production using correct recipe identifier
	Dim response as String
	response = Eyeplus.Execute("start production 12345")
	If response = "200" Then
		'Start pick and place scenario
		Dim it as Integer
		For it = 0 to 2000
			'Move robot out of field of view
			Move P, @P J[out_of_view]

			'Get one part coordinates
			response = Eyeplus.Execute("get_part")
			If Mid(response, 0, 3) = "200" Then
				'Set pick position
				Call SetPositionNumber(42)

				'Move on the part
				Approach P, P[42], 50
				Move L, P[42]

				Reset IO[24] 'Close Gripper

				'Move on place position
				Depart 50
				Approach P, P[place_position], 50
				Move L, P[place_position]

				Set IO[24] ' Open Gripper

				'Move to above place position
				Depart 50
			End If
		Next
	End If

	' Stop EYE+ production state
	response = Eyeplus.Execute("stop production")

	' Stop communication
	cao.Controllers.Remove Eyeplus.Index
	Eyeplus = Nothing
End Sub

Sub SetPositionNumber(ByRef num as Integer)
	' Implement your method to extract the coordinates and set the desired point position
	' ...
End Sub
```