# Manual


# FlexiVision

## EPSON PLUG-IN

ars | Feeding Industrial Robotics

# INDEX



This Plugin was developed with the idea of communicating **quickly and safely with Flexivision 2.0** through **Epson** robots by using instructions in **RC+.**
The Plugin does not require any additional licence

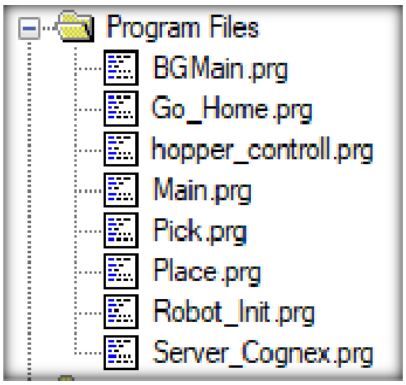
## FlexiBowl® Plug-In

EPSON®

# Plug-In Installation

## Introduction.

Together with this guide, a basic example developed directly with **RobotStudio** is provided in order to understand the steps to implement the application.




```
Function BGMain

    'start server
    If ((TaskState(Server_Cognex) <> 1) And (TaskState(Server_Cognex) <> 2)) Then
    Xqt Server_Cognex
    EndIf


Fend
```

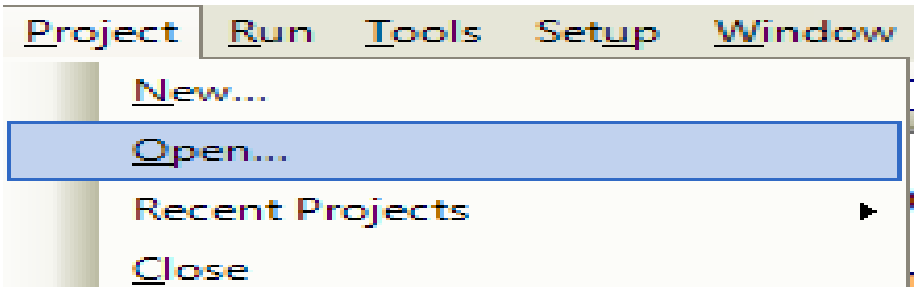
We will use two tasks, one for the main cycle and one parallel (**BGMain**) that will manage communication with **FlexiVision** without ever being interrupted by stops or emergencies.

# Plug-In Installation

## Step 1.


Copy the Plug-in provided by ARS to the ***projects*** folder of the ***RC+*** program

## Step 2.




In the dialogue window that will appear, **Open** the Project of the Plug-in previously imported.

## Step 3.

```
'server setup & start
SetNet #201, "169.254.75.100", 8888,
```

After importing the **FlexiVision Plug-in**, simply modify this line of code by setting the **IP** of your PC (where **FlexiVision** is installed) and the port you intend to use.

## Step 4.

```
MemOn Semaforo_Cognex;
```

The "**Semaforo_Cognex**" signal will allow FlexiVision to take a photo. The latter will be set to **OFF** when a removable item is identified by vision.

# FlexiVision Command List

To send the command to FlexiVision you must modify the value of the "command" string.

| N_Mission | Command | Action |
|---|---|---|
| **1** | "start_Locator" | Starts the parts localization process by recalling the FlexiBowl handling routine in case there are no parts that can be picked up. ***Return:*** "Pattern1;x;y;r". |
| **2** | "stop_Locator" | Stops the process of locating the object with the aid of the FlexiBowl. |
| **3** | "turn_Locator" | If no parts are picked up, by this command the operator can make the Flexibowl rotate and the "start_Locator" routine start. ***Return:*** "Pattern1;x;y;r". |
| **4** | "test_Locator" | Starts the process of locating the object without the aid of the FlexiBowl. ***Return:*** "Pattern1;x;y;r". |
| **5** | "start_Control" | Starts the inspection cycle. ***Return:*** "Control1;x;y;r". |
| **6** | "state_Locator" | Locator status diagnostics is shown: ***Return:*** "Locator is Running" "Locator is in Error" "Locator is not Running". |
| **7** | "start_Empty" | Start the FlexiBowl® Quick-Emptying sequence. ***Return:*** "start_Empty ended" |
| **8** | "get_Recipe" | The name of the recipe currently loaded on FlexiVision is shown. ***Return:*** "recipe name". |
| **9** | "set_Recipe=recipe name" | The recipe corresponding to the sent "recipe name" is loaded. |

# Script

```
Global String Data_Recived_From_Cognex$
Global String Data_Recived_From_Cognex_cnt$
Global Integer hopper_number
Global Integer Hopper_Array_Pass(2)
Global Integer contapassi
Global String recipe$
Global String recipe_name$
Global Double tempo_risposta
String Command$
Function Server_Cognex()

'in case of error restart from top
Inizio:
Integer errNum
'OnErr GoTo Errhandler

'position variables
Real x, y, z, roll, v, w

CloseNet #201
Wait 0.1

'Program variables
MemOn Semaforo_Cognex
String Data_Send_To_Cognex$
String Data_Rcv_Splitted$(0)

'server setup & start
SetNet #201, "169.254.75.100", 8888, CR, NONE, 0
OpenNet #201 As Server
Print "Server Cognex  started"

'wait for connection
WaitNet #201
Print "Connected Cognex "

'wait FlexiVision boot up
Wait 100

'set recipe "example"
Call ChangeRecipe("example")

'inizialize the first command to be sent to FlexiVision
Command$ = "start_Locator"
TmReset 7
```

FlexiBowl by ars

# Script

```
'inizialize the first command to be sent to FlexiVision
Command$ = "start_Locator"
TmReset 7

'Main cycle
Do While True

'wait free to snap
waitcommand:
Wait MemSw(Semaforo_Cognex) = On Or MemSw(Semaforo_ControllCam) = On

'if part is requested from the robot
If MemSw(Semaforo_Cognex) = On Then

resnap:

    'send job to be executed by cognex
    Print #201, Command$
    TmReset 5

    'read data received from Cognex
    Input #201, Data_Recived_From_Cognex$
    tempo_risposta = Tmr(5)

    'print data received from Cognex
    Print "Server received from Cognex :", Data_Recived_From_Cognex$

    'check if the hopper should be activated
    If (InStr(Data_Recived_From_Cognex$, "Hopper") >= 0) Then

        'activate the hopper
        If ((TaskState(hopper_controll) <> 1) And (TaskState(hopper_controll) <> 2)) Then
        Xqt hopper_controll
        EndIf
        Command$ = "hopper_Done"

    'check if flexivision it's in error state
    ElseIf (InStr(Data_Recived_From_Cognex$, "#") >= 0) Then

    'restart the server
    GoTo inizio

    'if Data_Recived_From_Cognex$ contains "Pattern" split the picking position
    ElseIf (InStr(Data_Recived_From_Cognex$, "Pattern") >= 0) Then

    'prepare the new command
    Command$ = "start_Locator"

    'if 1 or more items found split string
    ParseStr Data_Recived_From_Cognex$, Data_Rcv_Splitted$(), ";"
```

FlexiBowl by ars

# Script

```
'inizialize the first command to be sent to FlexiVision
Command$ = "start_Locator"
TmReset 7

'Main cycle
Do While True

'wait free to snap
waitcommand:
Wait MemSw(Semaforo_Cognex) = On Or MemSw(Semaforo_ControllCam) = On

'if part is requested from the robot
If MemSw(Semaforo_Cognex) = On Then

resnap:

    'send job to be executed by cognex
    Print #201, Command$
    TmReset 5

    'read data received from Cognex
    Input #201, Data_Recived_From_Cognex$
    tempo_risposta = Tmr(5)

    'print data received from Cognex
    Print "Server received from Cognex :", Data_Recived_From_Cognex$

    'check if the hopper should be activated
    If (InStr(Data_Recived_From_Cognex$, "Hopper") >= 0) Then

        'activate the hopper
        If ((TaskState(hopper_controll) <> 1) And (TaskState(hopper_controll) <> 2)) Then
        Xqt hopper_controll
        EndIf
        Command$ = "hopper_Done"

    'check if flexivision it's in error state
    ElseIf (InStr(Data_Recived_From_Cognex$, "#") >= 0) Then

    'restart the server
    GoTo inizio

    'if Data_Recived_From_Cognex$ contains "Pattern" split the picking position
    ElseIf (InStr(Data_Recived_From_Cognex$, "Pattern") >= 0) Then

    'prepare the new command
    Command$ = "start_Locator"

    'if 1 or more items found split string
    ParseStr Data_Recived_From_Cognex$, Data_Rcv_Splitted$(), ";"
```

# Script

```
        'return in the main cycle
        GoTo waitcommand

    EndIf

Loop
Exit Function

        '****ERROR HANDLER******
Errhandler:
        'print error msg and restart the server
        CloseNet #201
        errNum = Err
        Print "The error code is ", errNum
        Print "The error message is ", ErrMsg$(errNum)
        EResume Inizio

Fend
```

```
Function ChangeRecipe(recipe_from_plc$ As String)

String recipe_from_cognex$

'loop untill FlexiVision's recipe is not equal to the correct one
Do While (Val(recipe_from_cognex$) <> Val(recipe_from_plc$))

    'set the recipe to FlexiVision
    Print #201, "set_Recipe=" + recipe_from_plc$

    Wait 0.3

    'read the recipe from FlexiVision
    Print #201, "get_Recipe"

    'read data received from FlexiVision
    Input #201, Data_Recived_From_Cognex$

    'print data received from FlexiVision
    Print "Server received from Cognex :", Data_Recived_From_Cognex$

    recipe_from_cognex$ = Data_Recived_From_Cognex$

Loop

Fend
```

# Script

```
Fend
Function Controll_cam

    'trigger controll cam
    Print #201, "start_Control"

    'read data received from FlexiVision
    Line Input #201, Data_Recived_From_Cognex_cnt$


    'print data received from FlexiVision
    Print "Server received from Cognex :", Data_Recived_From_Cognex_cnt$

    'check if model is found
    If (InStr(Data_Recived_From_Cognex_cnt$, "Null") >= 0) Then

        MemOff CNT_Found

    Else

        MemOn CNT_Found

    EndIf

    Print #201, "cam_done"

Fend
```