Manual

FlexiVision

JAKA Robotics Plug-In

ars automation

INDEX

1. **Plug-In installation**
2. **How to use the plug-in**
3. **Plug-In functions**
    1. **Recipe Management Functions**
    2. **Vision and movement functions**
    3. **Movement functions**
4. **Code example**
5. **FlexiVision® Command List**

This Plug-In has been developed to streamline communication between **JAKA** robots and the FlexiVision® system.

With this integration, users can easily implement a **fast and secure interface** with the FlexiVision® vision system.

The Plug-In provides a stable and immediate connection, enabling the transmission of all necessary commands for the efficient operation of an application using the FlexiVision® system.

FlexiVision® Plug-In

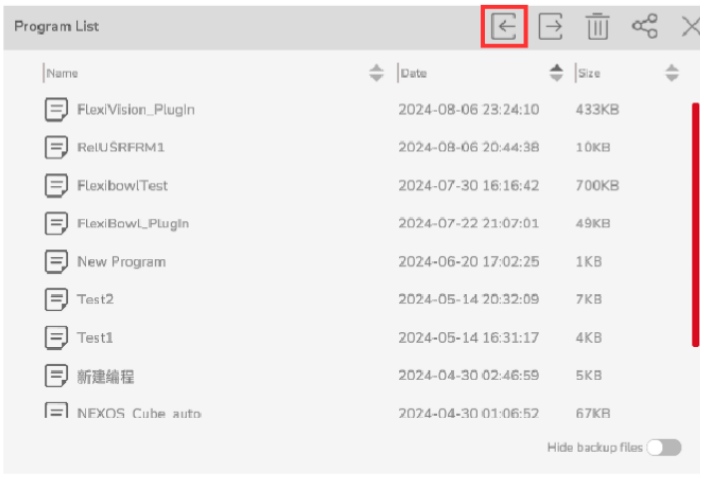
Plug-In installation

Before starting work with the plug-in, ensure that the "**Jaka zu**" software is installed and configured.
Once the software is open, connect via Wi-Fi or Ethernet cable through the appropriate LAN port.
Do not unzip the zip file called "FlexiVision_PlugIn.zip" .

Step 1.

Click on the folder icon on the right of the screen.

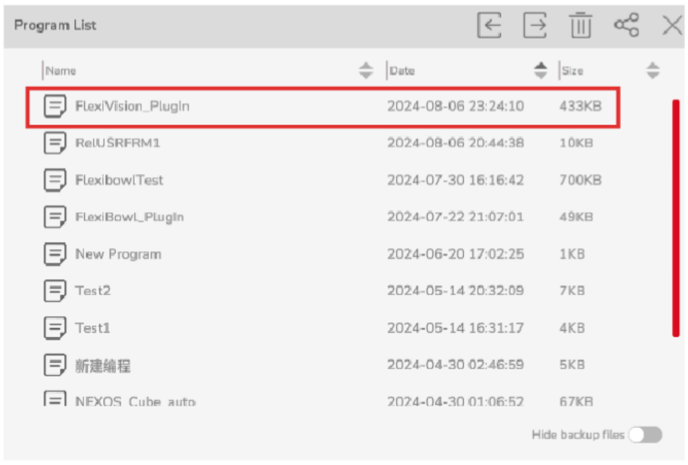
Step 2.



Name	Date	Size
FlexiVision_PlugIn	2024-08-06 23:24:10	433KB
RelUSRFRM1	2024-08-06 20:44:38	10KB
FlexibowlTest	2024-07-30 16:16:42	700KB
FlexiBowl_PlugIn	2024-07-22 21:07:01	49KB
New Program	2024-06-20 17:02:25	1KB
Test2	2024-05-14 20:32:09	7KB
Test1	2024-05-14 16:31:17	4KB
新建编程	2024-04-30 02:46:59	5KB
NEXOS_Cube_auto	2024-04-30 01:06:52	67KB

Before proceeding, click the **"import button"** (as shown in the image), navigate through the folders, and select the **FlexiVision_PlugIn**.

Step 3.

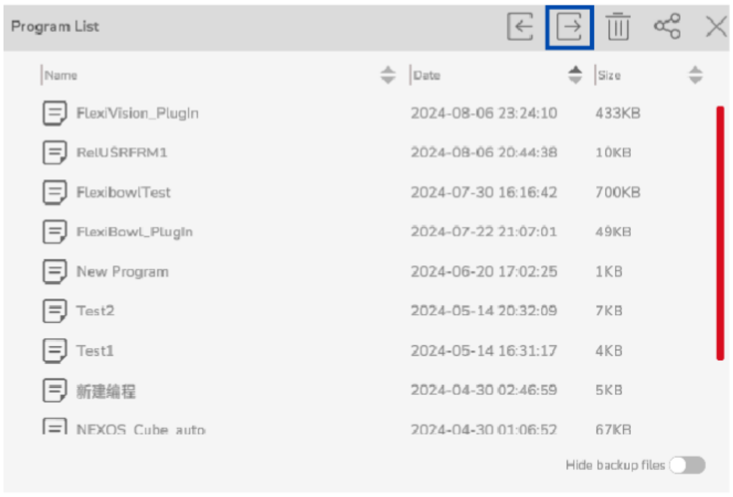


Name	Date	Size
FlexiVision_PlugIn	2024-08-06 23:24:10	433KB
RelUSRFRM1	2024-08-06 20:44:38	10KB
FlexibowlTest	2024-07-30 16:16:42	700KB
FlexiBowl_PlugIn	2024-07-22 21:07:01	49KB
New Program	2024-06-20 17:02:25	1KB
Test2	2024-05-14 20:32:09	7KB
Test1	2024-05-14 16:31:17	4KB
新建编程	2024-04-30 02:46:59	5KB
NEXOS_Cube_auto	2024-04-30 01:06:52	67KB

Click on the name of the plug-in you want to open.

Plug-In installation

Step 4.



Program List

Name	Date	Size
FlexiVision_PlugIn	2024-08-06 23:24:10	433KB
RelUSRFRM1	2024-08-06 20:44:38	10KB
FlexibowlTest	2024-07-30 16:16:42	700KB
FlexiBowl_PlugIn	2024-07-22 21:07:01	49KB
New Program	2024-06-20 17:02:25	1KB
Test2	2024-05-14 20:32:09	7KB
Test1	2024-05-14 16:31:17	4KB
新建编程	2024-04-30 02:46:59	5KB
NEXOS_Cube_auto	2024-04-30 01:06:52	67KB

Hide backup files

If you wish to export the plug-in after making changes, click the **"Export button"** and save it to your preferred location.

Plug-In installation

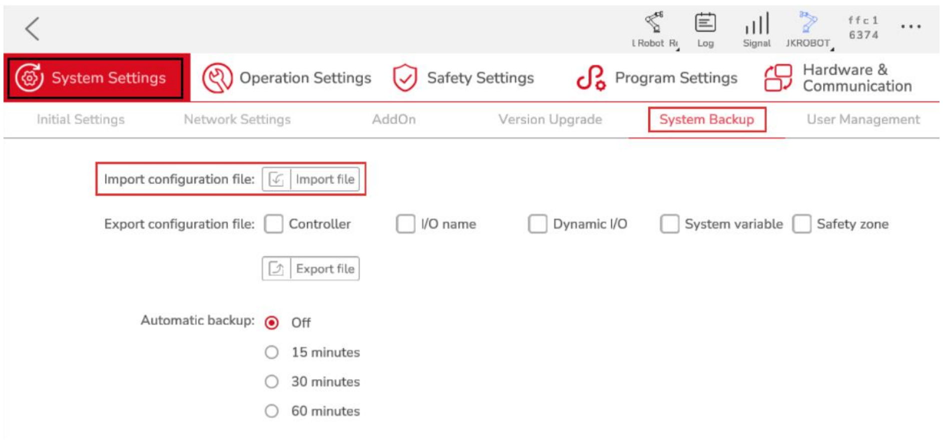
We will now be working with the file **"zucsettings.tar.gz"**, which is included in the zip folder we sent to you.
Please note that you should not extract or modify this file in any way.
This file contains all the system settings for the robot, which you will need to import into your robot as part of the setup process.

Step 5.

Click on the **Settings icon** on the top-right of the screen.

Settings

Step 6.



Import configuration file: Import file

Export configuration file: ☐ Controller ☐ I/O name ☐ Dynamic I/O ☐ System variable ☐ Safety zone

Automatic backup: [✓ Off] 15 minutes 30 minutes 60 minutes

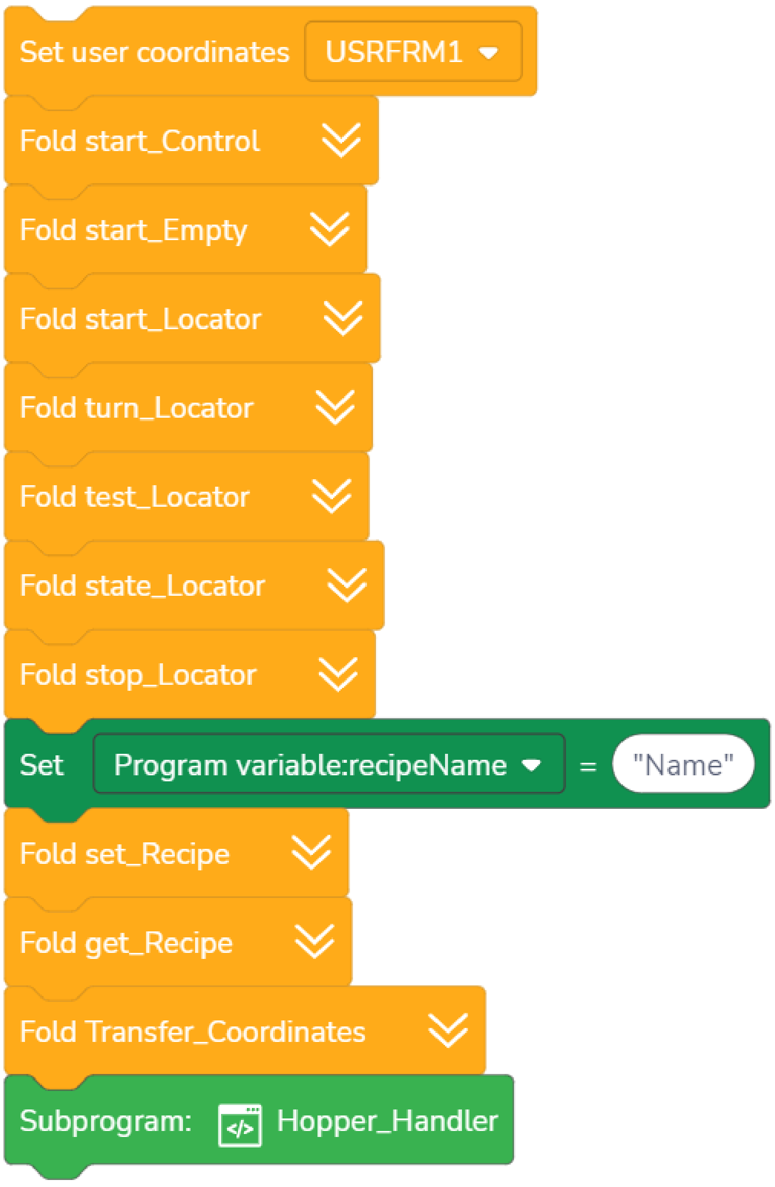
Enter the **"System Settings"** section, than go in the **"System Backup"** section and click on **"Import file"**

Step 7.

Please navigate to the correct directory on your computer, then select the correct file to successfully import it into the robot.

How to use the plug-in

How to use the functions correctly.



```
Set user coordinates  USRFRM1
Fold start_Control
Fold start_Empty
Fold start_Locator
Fold turn_Locator
Fold test_Locator
Fold state_Locator
Fold stop_Locator
Set  Program variable:recipeName  =  "Name"
Fold set_Recipe
Fold get_Recipe
Fold Transfer_Coordinates
Subprogram:  Hopper_Handler
```

For proper use of the plug-in, utilize the blocks available in the left section of the Jaka program.
Select the desired command, right-click on it, and choose **"Copy"**.
If any unwanted blocks have been copied, remove them by dragging them to the bottom center of the screen, over the trash can icon or right-click on it and than choose **"Delete"**.
For a detailed description of the various functions, please refer to the next chapter, **"Plug-In Functions"**.

Plug-In functions

1. Recipe Management Functions

1.1 FlexiVision – Get Recipe

The Get_Recipe function allows the user to output the name of the recipe currently loaded in the FlexiVision interface.
The recipe name is saved in the designated variable **"current_recipe"**.

Function

`null Get_Recipe()`

Return Value

`Current_reipe` = Value.

1.2 FlexiVision – Set Recipe

The Set_Recipe function allows the user to change the recipe from within the program.
Before sending the Set_Recipe function make sure to change the value of **"recipe_name"** to the recipe name you want to load.

Function

`null set_Recipe(string recipe_Name)`

Parameters

`string recipe_Name` – The recipe name you want to load in the FlexiVision Interface.

Plug-In functions

2. Vision and movement functions

2.1 FlexiVision – Start Empty

The Start_Empty function initiates a cleaning protocol by opening a small port on the FlexiBowl and commencing a rapid rotation sequence. This automated process efficiently cleans the FlexiBowl of all objects, eliminating the need for operator intervention.

Function

```
null start_Empty()
```

2.2 FlexiVision – Start Control

The Start Control function initiates the activation procedure for the control camera. The response is customizable, but to be interpreted by the plug-in, it must be in the following format: ControlN;x;y;rz. (Example: "Control1;3;4;5").

Function

```
null start_Control()
```

2.3 FlexiVision – Start Locator

The Start_Locator function moves the FlexiBowl using the saved sequence until the object is detected and then returns its coordinates and set on the Hopper_Flag if needed.

Function

```
null start_Locator()
```

2.4 FlexiVision – Turn Locator

The Turn Locator function moves the FlexiBowl® using the saved sequence, then takes a photo and checks if there is an object that the robot can pick up.

Function

```
null Turn Locator()
```

2.5 FlexiVision – Stop Locator

The Stop Locator function instantly terminates the object search process by FlexiVision®.

Function

```
null Turn Locator()
```

Movement functions

2.6 FlexiVision – State Locator

The State_Locator function returns the current state of FlexiVision.
Possible returned values:

- “Locator is Running”
- “Locator is in Error”
- “Locator is not Running”.

Funzione

```
Null state_Locator()
```

Return Value

`Current_State =` Value.

2.7 FlexiVision – Test Locator

The Test_Locator function runs a photo from the vision system and returns the coordinates of the object he found if there is at least one object pickable on the Flexibowl, if there are not returns Null;null;null;null

Funzione

```
CBUN_PCALL  MyDevice::start_Control(int n_FlexiVision)
```

Plug-In functions

3. Recipe Management Functions

3.1 FlexiVision – Transfer Coordinates

The Transfer_Coordinates function moves the robot above the object, using the Z value specified by the user. The function then approaches the object, and once in position, activates the suction device via the designated DO.

Funzione

`Null state_Locator()`

Return Value

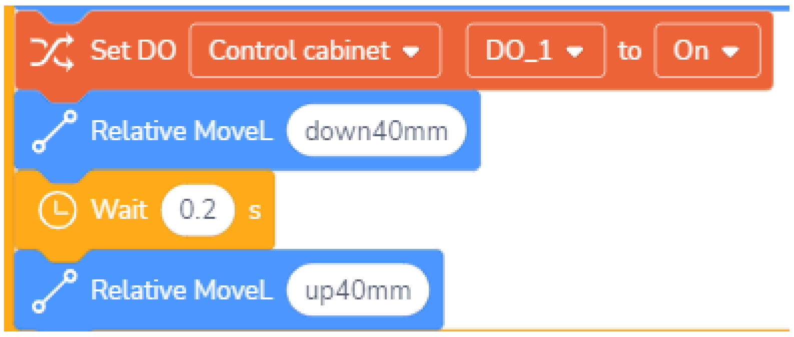
`Current_State` = Value.

What you need to do

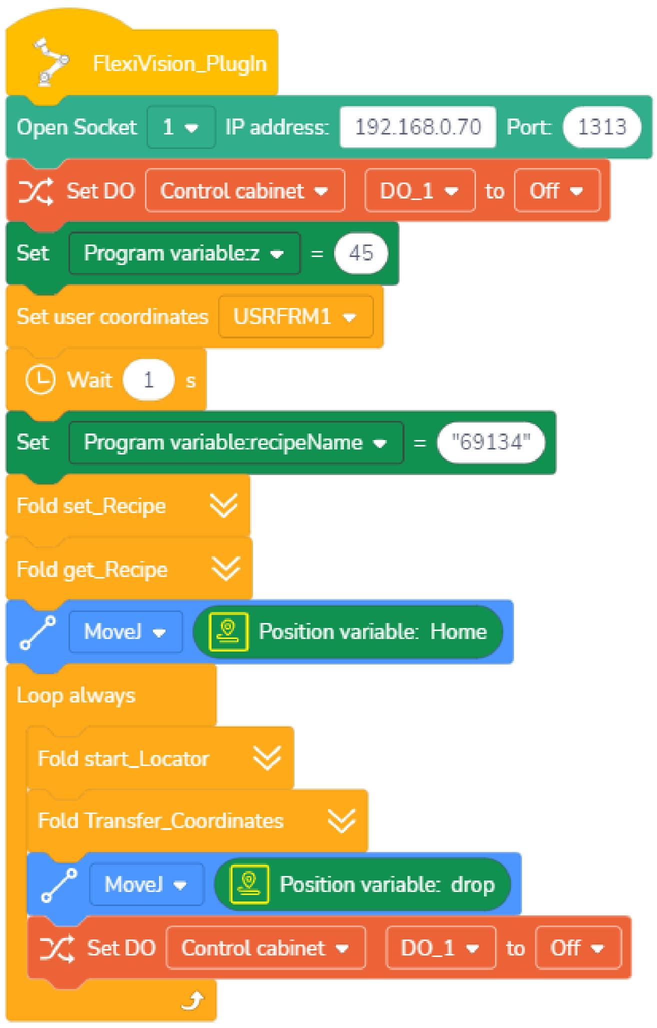
To use this function, you will need to customize the following parameters:

- DO (Digital Output): Replace the default DO with the specific DO corresponding to the suction device in your setup.
- "downXmm" and "upXmm" Strings: Modify these strings to reflect the desired approach height. The X value represents the height you want for the approach.

For example, if you need to pick an object at a height of 5mm and you want an approach of 40mm, you would set the Z value to 45mm, and use "down40mm" for the approach and "up40mm" for the retraction.



Code example



```
FlexiVision_PlugIn
Open Socket 1 ▾ IP address: 192.168.0.70 Port: 1313
Set DO Control cabinet ▾ DO_1 ▾ to Off ▾
Set Program variable:z ▾ = 45
Set user coordinates USRFRM1 ▾
Wait 1 s
Set Program variable:recipeName ▾ = "69134"
Fold set_Recipe
Fold get_Recipe
MoveJ ▾ Position variable: Home
Loop always
    Fold start_Locator
    Fold Transfer_Coordinates
    MoveJ ▾ Position variable: drop
    Set DO Control cabinet ▾ DO_1 ▾ to Off ▾
```

This code example demonstrates the implementation of a pick-and-place program.Initially, the program connects to the appropriate FlexiVision interface. We then set the desired approach z value, which in this case is pick_z + 40. The vacuum suction (DO) is turned off, and the appropriate User Frame is selected.Next, the variable recipeName is assigned the desired recipe name to be loaded into FlexiVision, followed by executing the set_Recipe command.

For debugging purposes, the get_Recipe command is then issued to verify the recipe loaded.The program proceeds with the robot's movements.

First, it moves to the home position, from which the start_Locator command is issued to retrieve coordinates from FlexiVision.

The Transfer_Coordinates function is then used to pick the object.

Finally, the robot moves to the designated drop position to release the object.

FlexiVision Command List

To send the command to FlexiVision, the value of the string "command" must be changed.

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.