ROBOTIQ

Robotiq Grippers for Universal Robots

Instruction Manual (Archives)

LEAN ROBOTICS

robotiq.com | leanrobotics.org

Revisions

Robotiq may modify this product without notice, when necessary, due to product improvements, modifications or changes in specifications. If such modification is made, the manual will also be revised, see revision information. See the latest version of this manual online at robotiq.com/support.

Revision 2019/12/19

- Official release

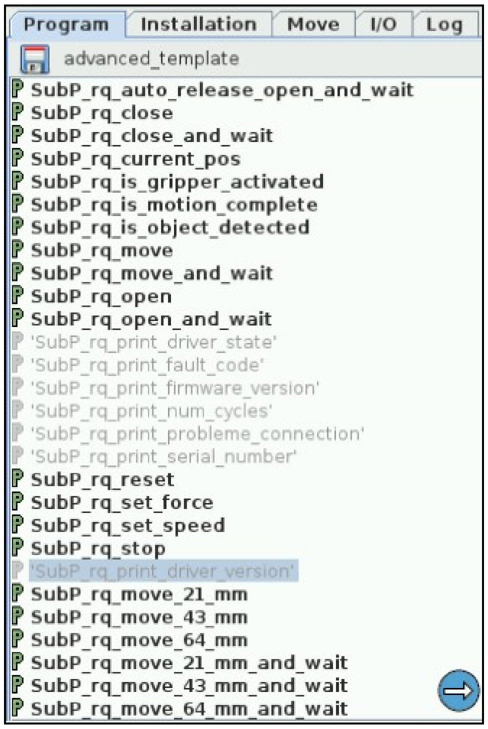
1. Control

1.1. Retro-compatibility of URCaps with legacy driver programs

The Gripper URCap can be used within programs that were made with the legacy driver package. To do so, you need to do the following modifications:

1. Open your .urp program in PolyScope and execute it.
    a. You should see an error message that tells you that some functions are double defined. This is because the URCap embeds the functions directly in the program preamble, so there is no need for a BeforeStart section.
2. Comment all the lines of the BeforeStart section from your older .urp program.
    a. You will still get an error. It arises because we changed some functions names due to a naming conflict.
3. To solve the problem, you can either suppress those functions or rename them.

Since they are probably not used in a production program, the fastest would be to suppress them, as shown in the figure below.



```
SubP_rq_auto_release_open_and_wait
SubP_rq_close
SubP_rq_close_and_wait
SubP_rq_current_pos
SubP_rq_is_gripper_activated
SubP_rq_is_motion_complete
SubP_rq_is_object_detected
SubP_rq_move
SubP_rq_move_and_wait
SubP_rq_open
SubP_rq_open_and_wait
'SubP_rq_print_driver_state'
'SubP_rq_print_fault_code'
'SubP_rq_print_firmware_version'
'SubP_rq_print_num_cycles'
'SubP_rq_print_probleme_connection'
'SubP_rq_print_serial_number'
SubP_rq_reset
SubP_rq_set_force
SubP_rq_set_speed
SubP_rq_stop
'SubP_rq_print_driver_version'
SubP_rq_move_21_mm
SubP_rq_move_43_mm
SubP_rq_move_64_mm
SubP_rq_move_21_mm_and_wait
SubP_rq_move_43_mm_and_wait
SubP_rq_move_64_mm_and_wait
```

Fig. 1-1: Older program's subprograms to suppress or rename to use with URCaps package.

Here is the list of the functions that were renamed:

- rq_print_fault_code -> rq_print_gripper_fault_code
- rq_print_num_cycles -> rq_print_gripper_num_cycles
- rq_print_driver_state -> rq_print_gripper_driver_state
- rq_print_firmware_version -> rq_print_gripper_firmware_version
- rq_print_driver_version -> rq_print_gripper_driver_version

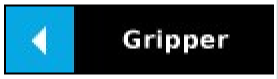
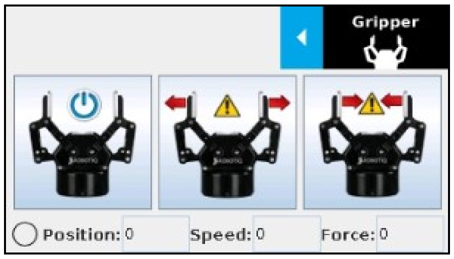
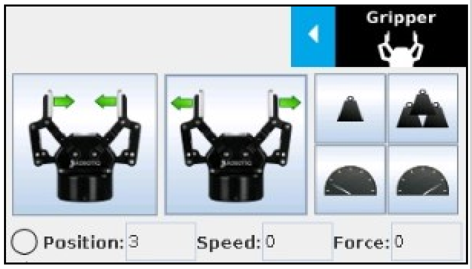
1.2. Control over Universal Robots without URCaps

If your Universal Robots controller is not compatible with the URCap package (see **Installation for Universal Robots** section) for compatibility), you can install the driver package. This package allows programming of the Gripper with scripts in a PolyScope program. It includes program templates and examples to help you get started with your own custom program. It also contains the Gripper toolbar for jogging and controlling the Gripper.

1.2.1. Gripper Toolbar

Overview

The Gripper toolbar is automatically installed with the driver package. The Gripper toolbar allows you to jog and test the Gripper. It is a great tool to try grasps with the Gripper while programming.

Toolbar collapsed	Toolbar expanded, Gripper activation window	Toolbar expanded, Gripper operation window :
		
Tap the Gripper button to expand the toolbar.	• When the Gripper is not activated, the toolbar shows this window. • You need to tap the Activate button to be able to jog the Gripper. • Emergency open and close allows you to control the Gripper without activating, this mode will use a very low speed and force setting	Use the buttons of this window to jog and test the Gripper.

Features

Toolbar collapsed

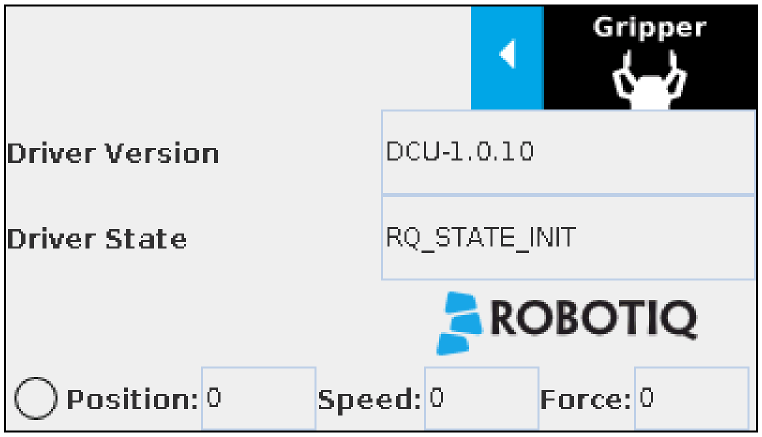
Icon	Functionality Name	Description
Gripper Gripper	Gripper toolbar	Tap to toggle between expand and collapse the Hand-E gripper toolbar. When grey, the functionality is not available.
	Activate	Tap to activate the Hand-E gripper. The Gripper will fully open and close to set the zero of the position value..
	Emergency open	Slowly moves the Gripper to its fully closed position.
	Emergency close	Slowly moves the Gripper to its fully open position.

Toolbar expanded

Icon	Functionality Name	Description
Gripper Gripper	Gripper toolbar	Tap to toggle between expand and collapse the Hand-E gripper toolbar. When grey, the functionality is not available.
	Open	Tap to fully open the Gripper.

Icon	Functionality Name	Description
	Close	Tap to fully close the Gripper.
Position	Requested position	Shows the actual position of the Gripper: • 0% : fully open • 100% : fully closed
Speed	Requested speed	Shows the actual speed set of the Gripper: • 0% : minimum speed • 100% : maximum speed
Force	Requested force	Shows the actual force set of the Gripper: • 0% : minimum force, regrasp feature disabled • 1% : minimum force, with regrasp feature enabled • 100% : maximum force, with regrasp feature enabled
	Increase force	Tap to increase the force request.
▲	Decrease force	Tap to decrease the force request.
	Increase speed	Tap to increase the speed request..
	Decrease speed	Tap to decrease the speed request.
○	No object detected	Icon shown when no object is detected during a grasp.
🟡	Object detected	Icon shown when an object is detected during a grasp.

When communication with the Gripper is not established, the expanded toolbar shows the driver version:



Driver Version DCU-1.0.10

Driver State RQ_STATE_INIT

Position: 0 Speed: 0 Force: 0

Fig. 1-2: Gripper toolbar expanded with driver version

Info

The driver state "RQ_STATE_INIT" means the driver is attempting to connect to a Robotiq Gripper. When connection is established, the normal toolbar detailed above will appear.

Tip

If you see the following toolbar with communication not established, check if your Gripper is powered first, then check if the RS-485 to USB converter is properly wired.

1.2.2. Demo Scripts

The following section details the demo scripts provided with the driver package.

pick_and_place_demo_with_subprograms.script is a demo script for pick and place applications using provided subprograms. The script uses subprograms included in the package such as **rq_set_force, rq_set_speed**, etc.

Info

All provided subprograms are identified with the prefix **rq_**.

The script uses **rq_speed** and **rq_force** as speed and force parameters to be used during the program. They can be modified using values from 0 to 255 (please refer to the **Provided Variables and Functions** section).

The script executes the following actions in sequence :

- Assign initial values to the global variables in the **Init Variables** section.
- Initiate the communication with the Gripper in the **BeforeStart** section.
- The **Robot Program** section contains the commands sent to the Gripper:
  - Activate the Gripper with **SubP_rq_activate_and_wait**.

Info

Remember that all **_and_wait** subprograms will wait for the action to be completed before going to the next step.

- Move the robot to a predetermined position

Tip

Run with Universal Robots simulator first or make sure that the UR robot work area is totally cleared before running the script, as it will move the robot.

- Close the Gripper with rq_close_and_wait.
- Watch for object detection status:
  - If an object is detected, the script moves the robot and opens the Gripper.
  - If no object is detected, the script prompts a warning

.pick_and_place_demo_async_partial_opening_without_subprograms.script is similar to the previous demo, but without using subprograms. This demo uses asynchronous commands so that the robot and the Gripper will move at the same time (the previous example had the Gripper and the robot move separately).

1.2.3. Custom Programs

You can create your own program that commands the Gripper with the provided templates :

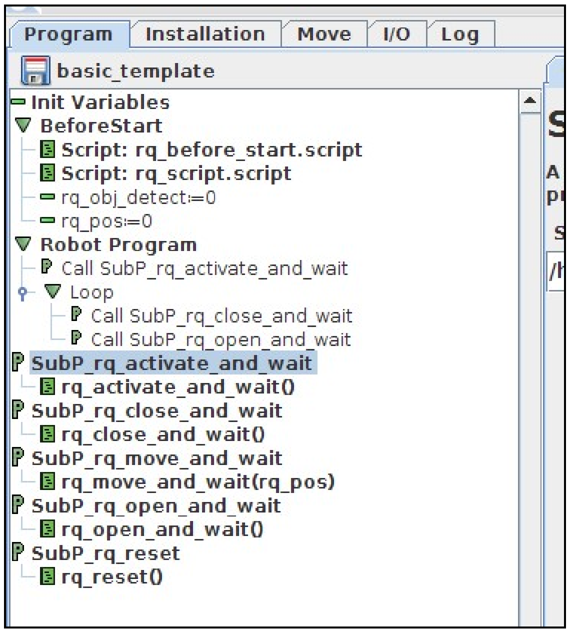
- Open basic_template.script in the list of provided templates;
- Push play to test the Gripper. The program will activate the Gripper and then do a loop of closing and opening the Gripper;
- Add your instructions under the robot program section. Program instructions can be added with PolyScope.

When programming an object pick up, use the **rq_is_object_detected subprogram** and **rq_object_detect** variable to know if an object has been picked. The subprogram sets the **rq_object_detect** variable to 1 if an object is detected, 0 otherwise.

Subprograms with the **_and_wait** will wait for the instruction to be completed before going to the next step. For example, **rq_close_and_wait** will wait for the motion to be completed before continuing to the next step, while **rq_close** will initiate motion and go to the next programmed step.

As shown in the figure below, the basic_template program will execute these instructions in a sequence:

- Assign initial values to the global variables with the Init Variables section.
- Initiate communication with the Gripper with the Before Start section.
- The Robot Program section contains the commands sent to the Gripper:
  - Activate the Gripper with SubP_rq_activate_and_wait.
  - Close the Gripper with rq_close_and_wait.
  - Open the Gripper with rq_open_and_wait.



Program | Installation | Move | I/O | Log

```
basic_template
Init Variables
▼ BeforeStart
    Script: rq_before_start.script
    Script: rq_script.script
    rq_obj_detect:=0
    rq_pos:=0
▼ Robot Program
    Call SubP_rq_activate_and_wait
    ▼ Loop
        Call SubP_rq_close_and_wait
        Call SubP_rq_open_and_wait
SubP_rq_activate_and_wait
    rq_activate_and_wait()
SubP_rq_close_and_wait
    rq_close_and_wait()
SubP_rq_move_and_wait
    rq_move_and_wait(rq_pos)
SubP_rq_open_and_wait
    rq_open_and_wait()
SubP_rq_reset
    rq_reset()
```

Fig. 1-3: Basic template as shown in UR PolyScope.

When using the **advanced_template**, you have access to all of the subprograms listed in the **Provided Variables and Functions** section. Unused subprograms can be removed from the list. You must use global variables to pass information, not arguments.

1.2.4. Provided Variables and Functions

The file **rq_script.script** contains function definitions and variables that enhance the programming of the Gripper. Here is the list of these variables and functions.

Variables are used when programming the Gripper using subprograms. Since it is not possible to pass arguments to subprograms, the global variables listed below must be used.

Name	Range	Description
rq_force()	[0-255]	Force set point. Change this variable by calling the subprogram **SubP_rq_set_force**.
rq_gripper_act()	[0-1]	1 if the Gripper is activated. This variable is updated by calling either **SubP_rq_is_gripper_activated or SubP_rq_activate_and_wait**.
rq_move_complete()	[0-1]	1 if the motion is complete. This variable is updated by calling one of the following: • SubP_rq_move_and_wait; • SubP_rq_open_and_wait; • SubP_rq_close_and_wait; • SubP_rq_is_motion_complete.
rq_object_detect()	[0-1]	1 if an object is detected. This variable is updated by calling one of the following:

Name	Range	Description
		• **SubP_rq_move_and_wait;** • **SubP_rq_open_and_wait;** • **SubP_rq_close_and_wait;** • **SubP_rq_is_object_detected.**
rq_pos()	[0-255]	Position set point. This variable is updated by calling **SubP_rq_current_pos**.
rq_speed()	[0-255]	Speed set point. Change this variable by calling the subprogram **SubP_rq_set_speed**.

Name	Description
rq_activate()	Sends the Gripper activation command. If it is already activated, nothing happens. Note that the Gripper must be activated to complete any other operation. Program execution continues before the end of activation.
rq_activate_and_wait()	Sends the Gripper activation command. If it is already activated, nothing happens. Note that the Gripper must be activated to complete any other operation. Program execution waits for the activation.
rq_auto_release_close_and_wait()	Slowly moves the Gripper to its maximum closed position. The Gripper must be activated after this command. Meant for emergency procedures.
rq_auto_release_open_and_wait()	Slowly moves the Gripper to its maximum opened position. The Gripper must be activated after this command. Meant for emergency procedures.
rq_close()	Moves the Gripper its fully closed position.
rq_close_and_wait()	Moves the Gripper to its fully closed position and waits until the motion is completed to execute the next command.
rq_current_pos()	Sets global variable **rq_pos** to the current Gripper position and returns this value.
rq_is_gripper_activated()	Sets global variable **rq_gripper_act** to 1 if the Gripper is activated and returns True. Otherwise sets the variable to 0 and returns False.
rq_is_motion_complete()	Sets global variable **rq_mov_complete** to 1 if the Gripper motion is complete and returns True. Otherwise sets the variable to 0 and returns False.
rq_is_object_detected()	Sets global variable **rq_obj_detect** to 1 if the Gripper has detected an object and returns True. Otherwise, sets the variable to 0 and returns False.

Name	Description
rq_move_and_wait()	Moves the Gripper to the position defined by the argument and waits until the motion is completed.
rq_move()	Moves the Gripper to the position defined by the argument.
rq_open()	Moves the Gripper to its fully opened position.
rq_open_and_wait()	Moves the Gripper to its fully opened position and waits until the motion is completed.
rq_set_force()	Writes the value of rq_force into the force setting.
rq_set_speed()	Writes the value of rq_speed into the speed setting.