Tutorial : Control of a C-Model Gripper Using the Modbus RTU Protocol

Description: This tutorial explains how to use the "robotiq_c_model_control" and "robotiq_modbus_rtu" packages to control a C-Model Gripper configured with the Modbus TCP protocol. The C-Model Gripper has 2 fingers. Please visit this website for more information on the Robotiq Adaptive Robot Grippers.

Tutorial level: Beginner

1 - Prerequisites

This tutorial assumes that you have connected a C-Model Gripper via the serial connection with a serial to USB converter to a USB port of your computer. For more information on the Gripper installation please look at the Robotiq C-Model user manual. If you wish to connect the Gripper via the TCP protocol, please have a look at the tutorial on Control of an S-Model Gripper using the TCP Protocol. To test your Gripper and the serial interface you can try the windows driver as explained in the Robotiq software installation page.

Finally, make sure the external dependency for the package "robotiq_modbus_tcp" has been installed. The dependency is the python package pyModbus. On Ubuntu Precise (12.04), it is simply installed using:

```
$ rosdep install robotiq_modbus_tcp
```

On other systems, it can be installed using:

```
$ easy_install -U pymodbus
```

It is possible that you are required to have administrator rights. If so, simply add "sudo" at the beginning of the command.

2 - Configuring the Serial Port

To find out the port on which the controller is connected, use:

```
$ dmesg | grep tty
```

It's normally attributed to ttyUSB0, which we can then call with /dev/ttyUSB0. You might need to change the access permissions on the USB port. It can be done with:

```
$ sudo chmod 777 /dev/ttyUSB0
```

Once this is done, the driver's node can be started.

3 - Starting the Node

3.1 - Run the C-Model Driver Node

After sending the usual "roscore" command, the Gripper can be driven by the node "CModelRtuNode.py" contained in the package "robotiq_c_model_control". The USB port name has to be provided as an argument.

For example, the driver for controlling a C-Model Gripper with the USB port name /dev/ttyUSB0 is launched with the following command:

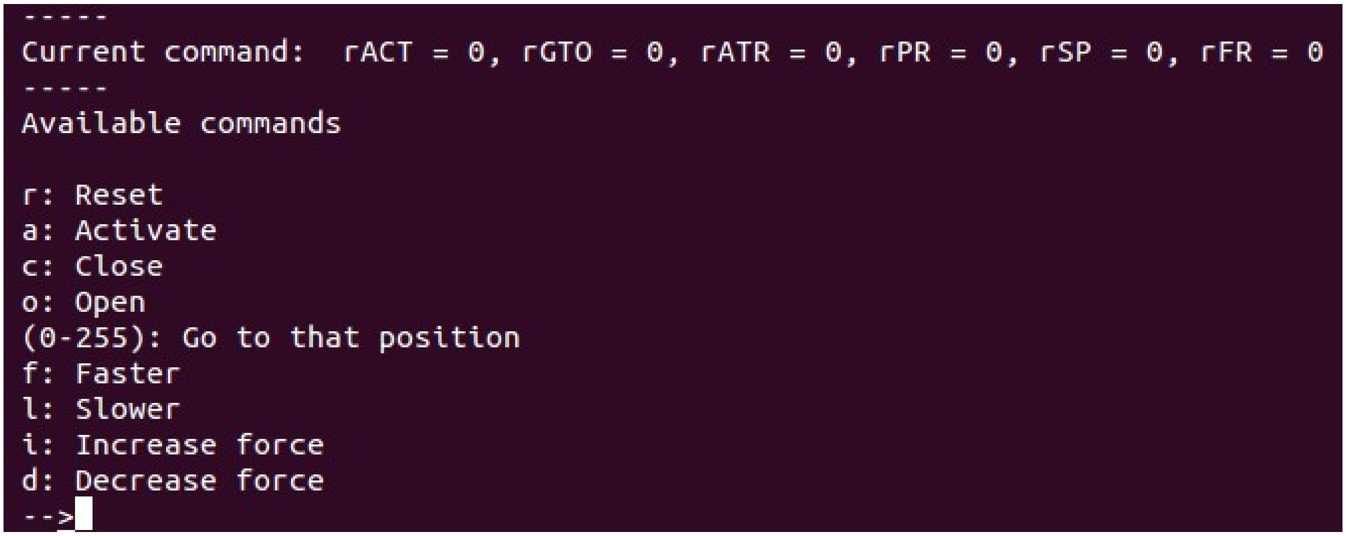
```
$ rosrun robotiq_c_model_control CModelRtuNode.py /dev/ttyUSB0
```

When the Gripper is successfully connected, the LED light appears blue.

3.2 - Run the C-Model Simple Controller Node

The driver listens for messages on "CModelRobotOutput" using the "SModel_robot_output" msg type. The messages are interpreted and commands are sent to the Gripper accordingly. A simple controller node is provided which can be run in another terminal, using the command:

```
$ rosrun robotiq_c_model_control CModelSimpleController.py
```



```
-----
Current command:  rACT = 0, rGTO = 0, rATR = 0, rPR = 0, rSP = 0, rFR = 0
-----
Available commands

r: Reset
a: Activate
c: Close
o: Open
(0-255): Go to that position
f: Faster
l: Slower
i: Increase force
d: Decrease force
-->
```

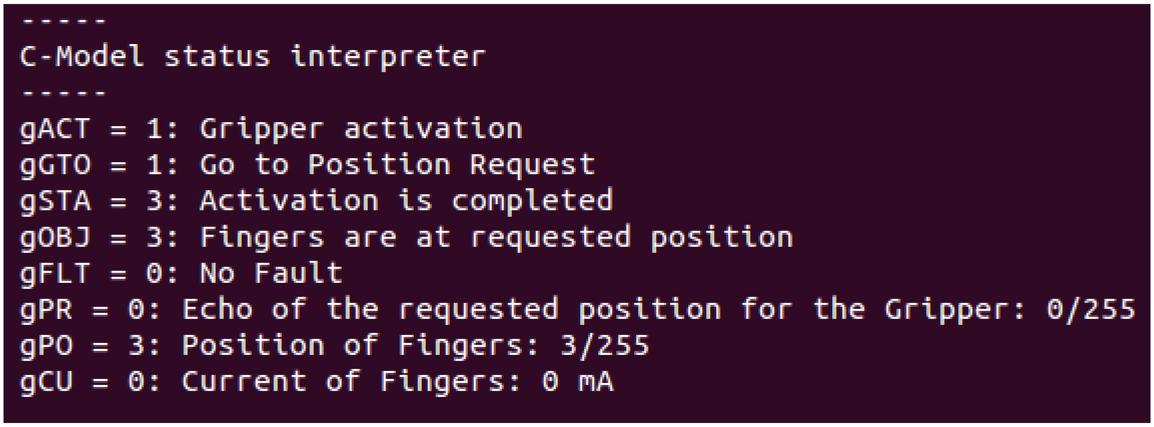
Figure 1 - Simple Controller Node

The "CModel_robot_output" msg type is simply composed of the robot output variables described in the Robotiq C-Model user manual. The simple controller node can therefore be modified to send custom commands to the Gripper.

3.3 - Run the C-Model Status Listener Node

In the package "robotiq_c_model_control", there is also a node for listening and interpreting the status of the Gripper. The driver publishes the status of the Gripper on "CModelRobotInput" using the "CModel_robot_input" msg type. The msg type is composed of the robot input variables described in the Robotiq C-Model user manual. The status listener node can be run in another terminal, using the following command:

```
$ rosrun robotiq_c_model_control CModelStatusListener.py
```



```
-----
C-Model status interpreter
-----
gACT = 1: Gripper activation
gGTO = 1: Go to Position Request
gSTA = 3: Activation is completed
gOBJ = 3: Fingers are at requested position
gFLT = 0: No Fault
gPR = 0: Echo of the requested position for the Gripper: 0/255
gPO = 3: Position of Fingers: 3/255
gCU = 0: Current of Fingers: 0 mA
```

Figure 2 - C-Model Status Listener