

How to control Universal Robot by using ROS2

Version: 1.0, 2021/05/18

Author: Qiu zhe

PREFACE

This introduction aims to provide a tutorial about how to control the real Universal Robot by using the Robot Operating System 2 (ROS2). The introduction is compatible across the entire line of UR robots -- from 3 kg to 16 kg payload and includes both the CB3 and the E-series. Especially, an ur_5e robot is used for explanation.

Content

1. Preliminaries

1.1 ROS distribution

1.2 Ubuntu system

2. Introduction

2.1 Download and build ROS 2 packages

2.2 Hardware setup: setting an ur_5e robot

2.2.1 Preparation

2.2.2 Install an URCap on an e-Series robot

3. Examples

3.1 Preparation

3.1.1 Set IP address of the robot

3.1.2 Set IP address of the control PC

3.2 Test joint trajectory controller

3.3 Test scaled_joint_trajectory_controller

3.4 Test MoveIt plugin

1. Preliminaries

1.1 ROS 2 distribution

ROS 2 Foxy is highly recommended.



(ROS 2 Foxy Fitzroy, released June 5th, 2020, supported until May 2023)

Installation: <https://docs.ros.org/en/foxy/Installation/Ubuntu-Development-Setup.html>

1.2 Ubuntu system

To match ROS 2 Foxy distribution, **Ubuntu 20.04** is required.

Installation: <https://ubuntu.com/download/desktop>

2. Introduction

This introduction is based on the official universal robot ROS 2 driver:
https://github.com/UniversalRobots/Universal_Robots_ROS2_Driver

2.1 Download and build ROS 2 packages

Follow the steps below to build the required ROS 2 packages:

Step 1:

```
# source global ROS 2
```

```
$ gedit ~/.bashrc
```

At the last line, add “source ~/ros2_foxy/install/local_setup.bash”

```
$ source ~/.bashrc
```

Step 2:

```
# create a new ROS 2 workspace
```

```
$ export COLCON_WS=~/.workspace/ros_ws_foxy_ur_driver
```

```
$ mkdir -p $COLCON_WS/src
```

Step 3:

```
# Pull relevant packages, install dependencies, compile, and source the workspace
```

```
$ cd $COLCON_WS
```

```
$ git clone
```

```
https://github.com/UniversalRobots/Universal\_Robots\_ROS2\_Driver.git  
src/Universal_Robots_ROS2_Driver
```

```
$ vcs import src --skip-existing --input
```

```
src/Universal_Robots_ROS2_Driver/Universal_Robots_ROS2_Driver.repos
```

```
$ rosdep install --ignore-src --from-paths src -y -r
```

```
$ colcon build --cmake-args -DCMAKE_BUILD_TYPE=Release
```

```
$ source install/setup.bash
```

Step 4:

```
# To use MoveIt, some additional packages should be added into workspace
```

```
$ cd $COLCON_WS
```

```
$ vcs import src --skip-existing --input
```

```
src/Universal_Robots_ROS2_Driver/MoveIt_Support.repos
```

```
$ vcs import src --skip-existing --input src/moveit2/moveit2.repos
```

```
$ rosdep install --ignore-src --from-paths src -y -r
```

```
$ colcon build --cmake-args -DCMAKE_BUILD_TYPE=Release
```

```
$ source install/setup.bash
```

2.2 Hardware setup: setting an ur_5e robot

2.2.1 Preparation

To enable external control of the UR robot from a control PC, you need to install the externalcontrol-1.0.5.urcap which can be found inside the resources folder of this driver:

(ros_ws_foxy_ur_driver→src→Universal_Robots_ROS2_Driver→ur_robot_driver→resources)

or download the latest from Universal_Robots_ExternalControl_URCap:

https://github.com/UniversalRobots/Universal_Robots_ExternalControl_URCap/releases

Note: For installing this URCap, a minimal PolyScope version 5.1 (for e-Series) is necessary.

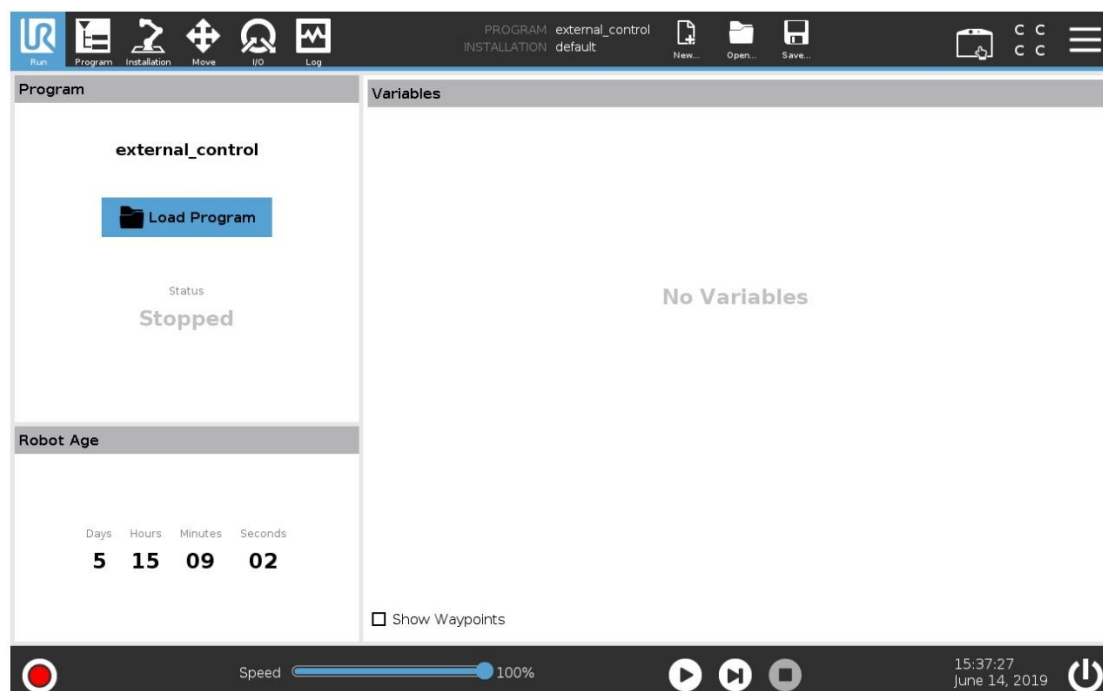
2.2.2 Install an URCap on an e-Series robot

For installing the necessary URCap and creating a program, please see the individual tutorial on how to setup a CB3 robot or [how to setup an e-Series robot](#)

To install it you first have to copy it to the robot's programs folder which can be done using a USB stick.

Step 1:

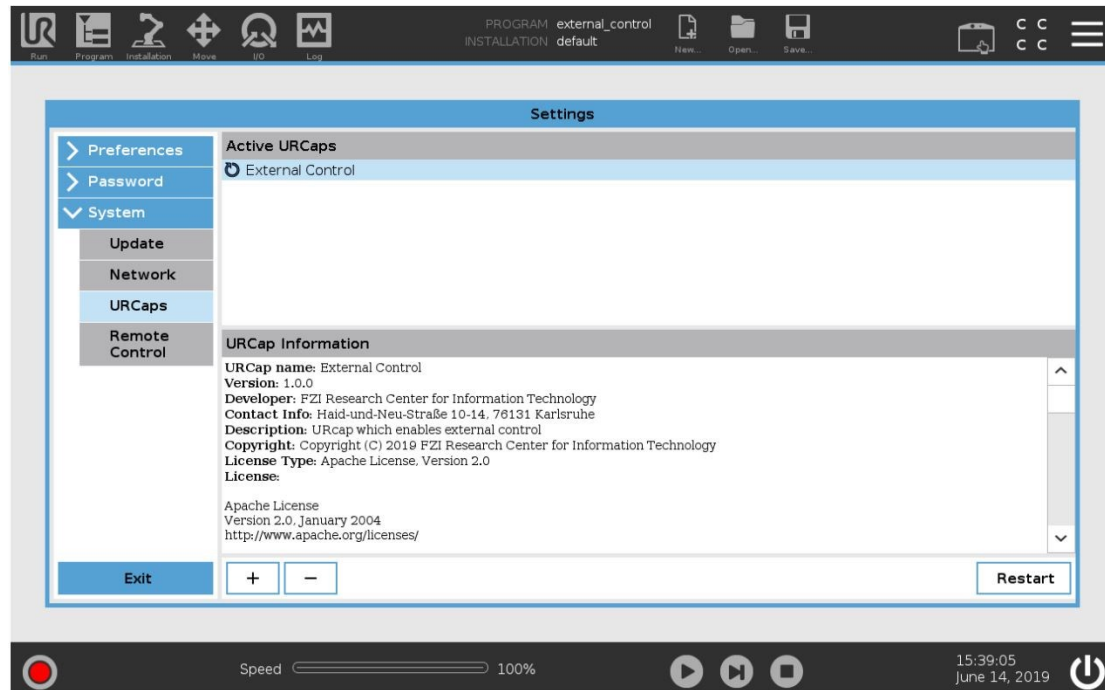
On the welcome screen, click on the hamburger menu in the top-right corner and select **Settings** to enter the robot's setup. Select **System** and then **URCaps** to enter the URCaps installation screen.



Step 2:

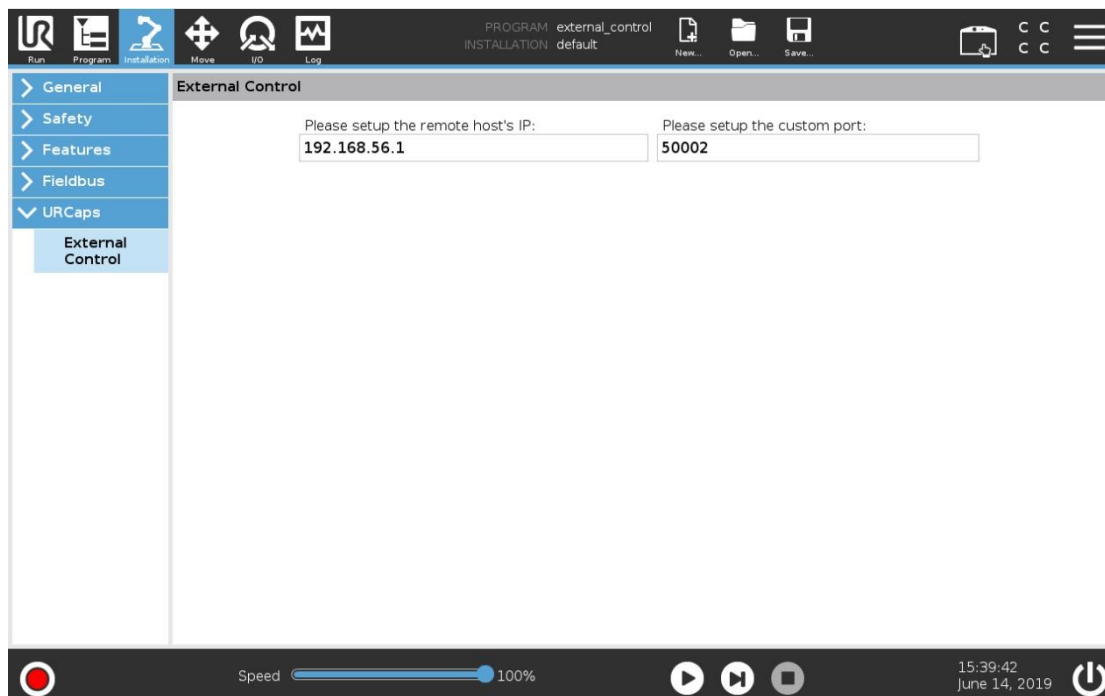
Click the little plus sign at the bottom to open the file selector. You should see all URCap files stored inside the robot's programs folder. Select and open

the `externalcontrol-1.0.5.urcap` file. Your URCaps view should now show the External Control in the list of active URCaps and a notification to restart the robot.



Step 3:

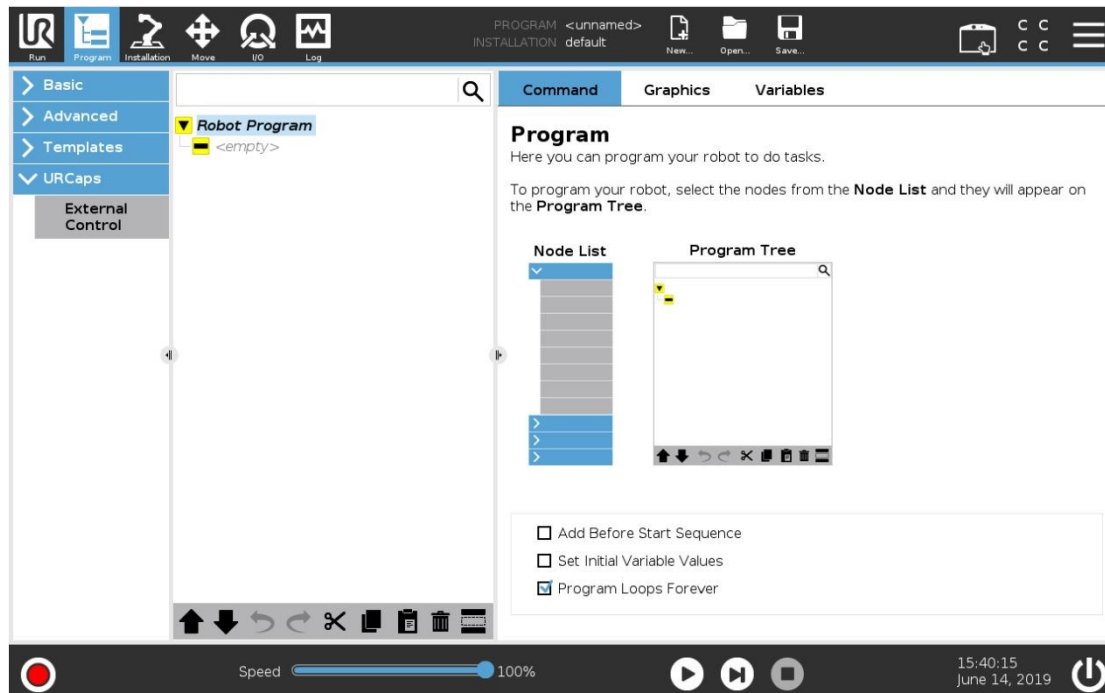
After the reboot you should find the External Control in URCaps tag inside Installation.



Step 4:

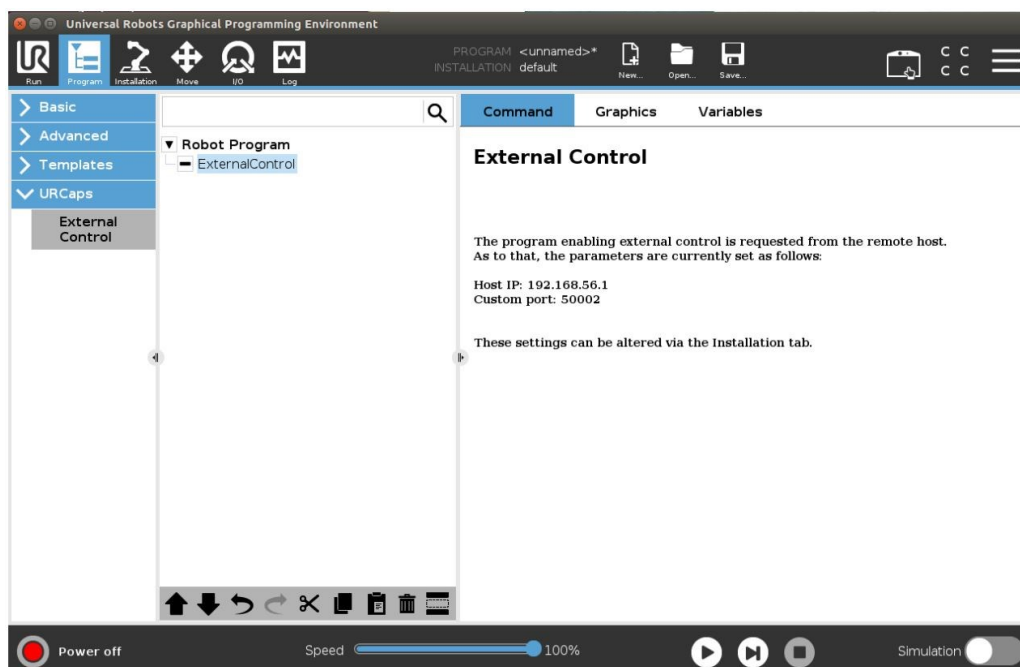
Here you should setup the IP address of the external PC which will be running the

ROS 2 driver. Note that the robot and the external PC have to be in the same network, ideally in a direct connection with each other to minimize network disturbances. The custom port should be left untouched for now.



Step 5:

To use the new URCaps, create a new program and insert the **External Control** program node into the program tree.



If you click on the **command** tab again, you'll see the settings entered inside the **Installation**. Check that they are correct, and then save the program. Your robot is now ready to be used together with this driver.

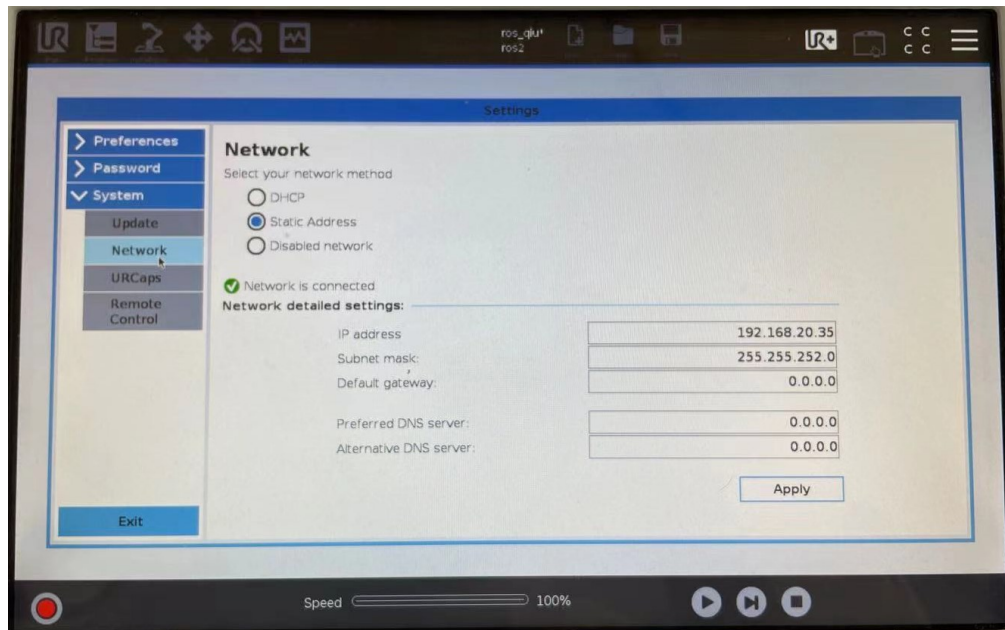
3. Examples

3.1 Preparation

First, the physical connection between the robot and the control PC should be established, e.g., connect the robot and the PC via a net cable.

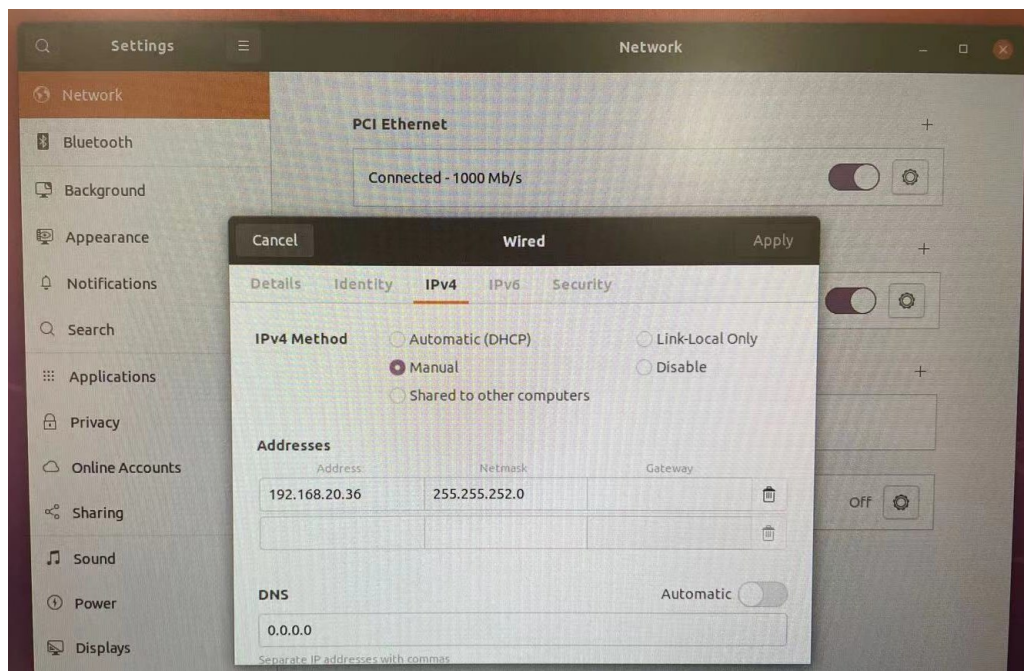
3.1.1 Set IP address of the robot

The IP address of the ur_5e robot is set as: 192.168.20.35.

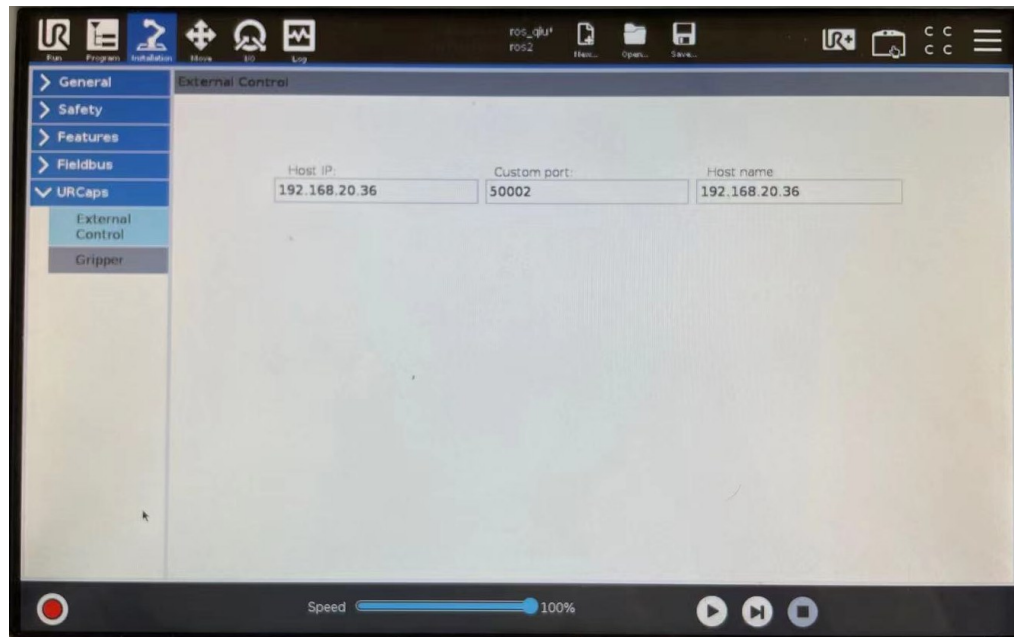


3.1.2 Set IP address of the control PC

First, configure the IP address of the control PC, which is set as: 192.168.20.36.



Second, update the information of External Control in URCaps in Installation.



As shown in the above figure, configure the Host IP and Host name.

3.2 Test joint trajectory controller

An ur_5e robot is controlled via `joint_trajectory_controller` by using a PC (ROS 2 Foxy with Ubuntu 20.04).

The following steps are recommended:

Step 1:

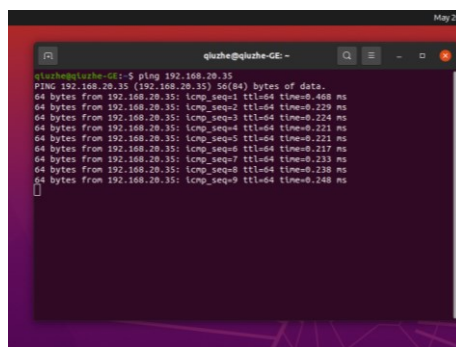
Power on the ur_5e robot, and confirm the connection between the robot and the PC.

The IP address of the ur_5e robot is set as: **192.168.20.35**.

The IP address of the PC is set as: **192.168.20.36**.

Open Terminal 1:

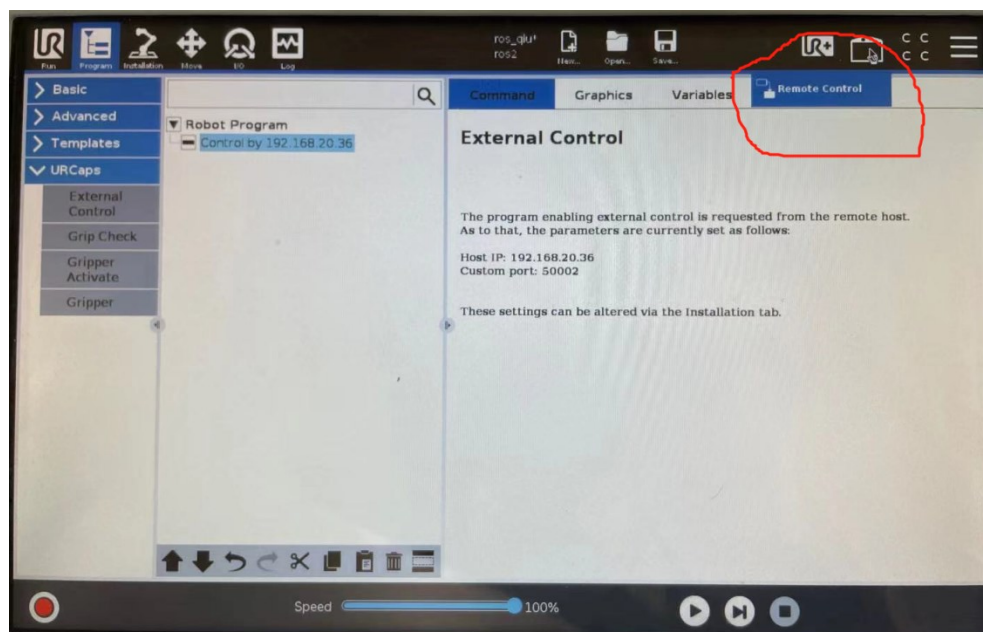
```
$ ping 192.168.20.35
```



We can see the robot and the PC are successfully connected.

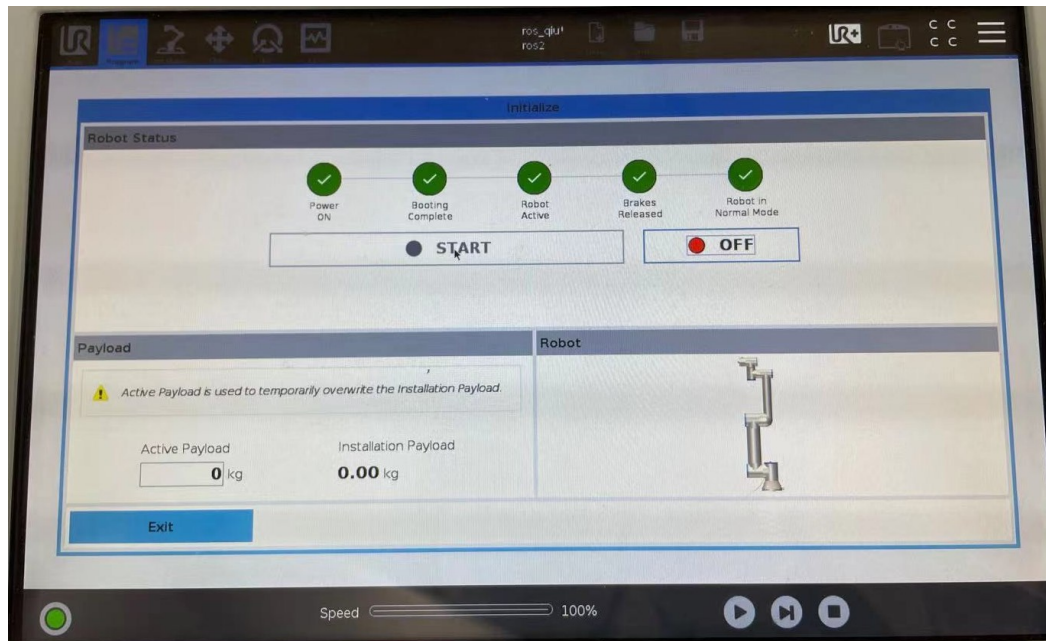
Step 2:

Set the robot control mode to the **local control mode** by using the teach-pendant.



As shown in the above figure, the local control mode has already set.

Step 3:
Start the robot.

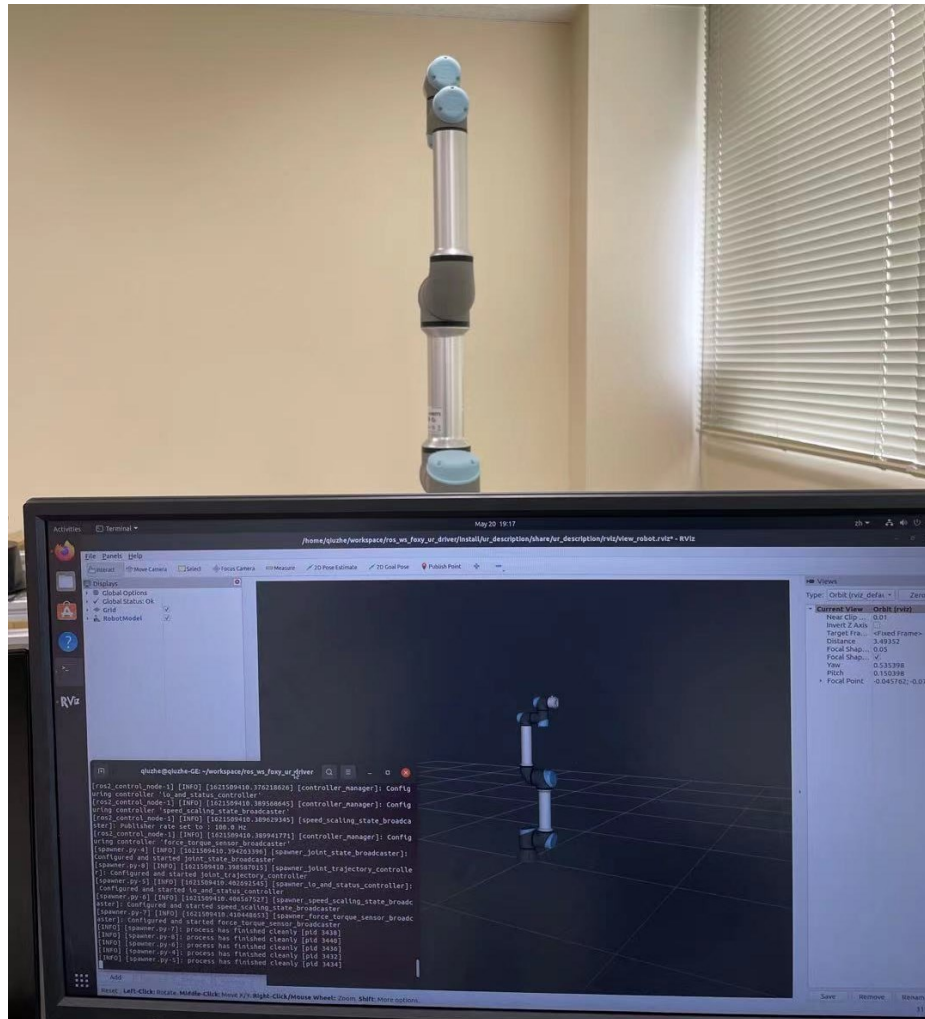


As shown in the above figure, the robot has already started.

Step 4:
Start the robot driver. Remember source the bash file first.
Open Terminal 2:
\$ export COLCON_WS=~/.workspace/ros_ws_foxy_ur_driver
\$ cd \$COLCON_WS
\$ source install/setup.bash

```
qluzhe@qluzhe-GE: ~/workspace/ros_ws_foxy_ur_driver
qluzhe@qluzhe-GE:~$ export COLCON_WS=~/.workspace/ros_ws_foxy_ur_driver
qluzhe@qluzhe-GE:~$ cd $COLCON_WS
qluzhe@qluzhe-GE:~/workspace/ros_ws_foxy_ur_driver$ source install/setup.bash
qluzhe@qluzhe-GE:~/workspace/ros_ws_foxy_ur_driver$
```

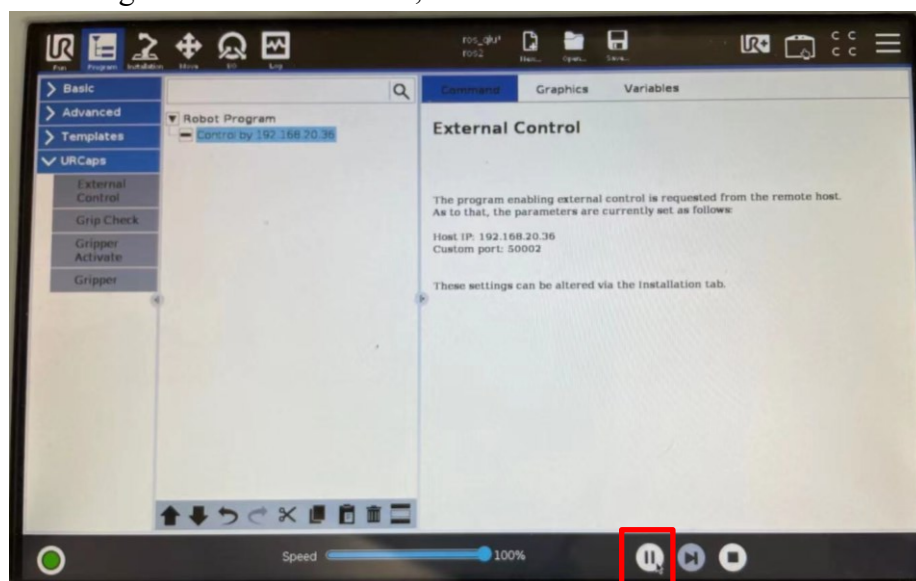
```
$ ros2 launch ur_bringup ur_control.launch.py ur_type:=ur5e
robot_ip:=192.168.20.35 launch_rviz:=true
```

As shown in the above figure, the driver is successfully started.

Step 5:

Load Robot Program: External Control, and start it.



As shown in the above figure, the program is already started.

Step 6:

Start the Joint Trajectory Controller. Remember source the bash file first.

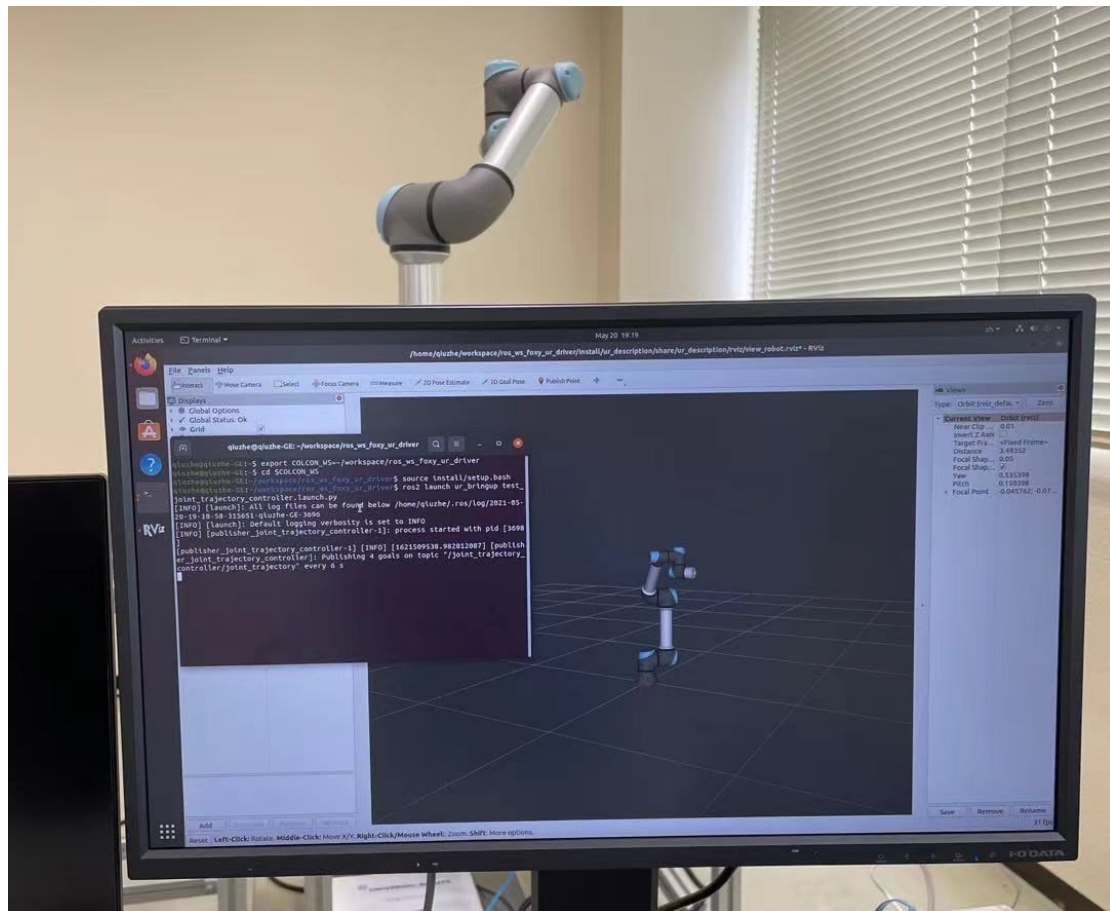
Open Terminal 3:

```
$ export COLCON_WS=~/.workspace/ros_ws_foxy_ur_driver
```

```
$ cd $COLCON_WS
```

```
$ source install/setup.bash
```

```
$ ros2 launch ur_bringup test_joint_trajectory_controller.launch.py
```



After a few seconds, the robot starts to move.

3.3 Test scaled_joint_trajectory_controller

An ur_5e robot is controlled via scaled_joint_trajectory_controller by using a PC (ROS 2 Foxy with Ubuntu 20.04).

The following steps are recommended:

Step 1:

Same as Step 1 of Example 3.2

Step 2:

Same as Step 2 of Example 3.2

Step 3:

Same as Step 3 of Example 3.2

Step 4:

Start the robot driver. Remember source the bash file first.

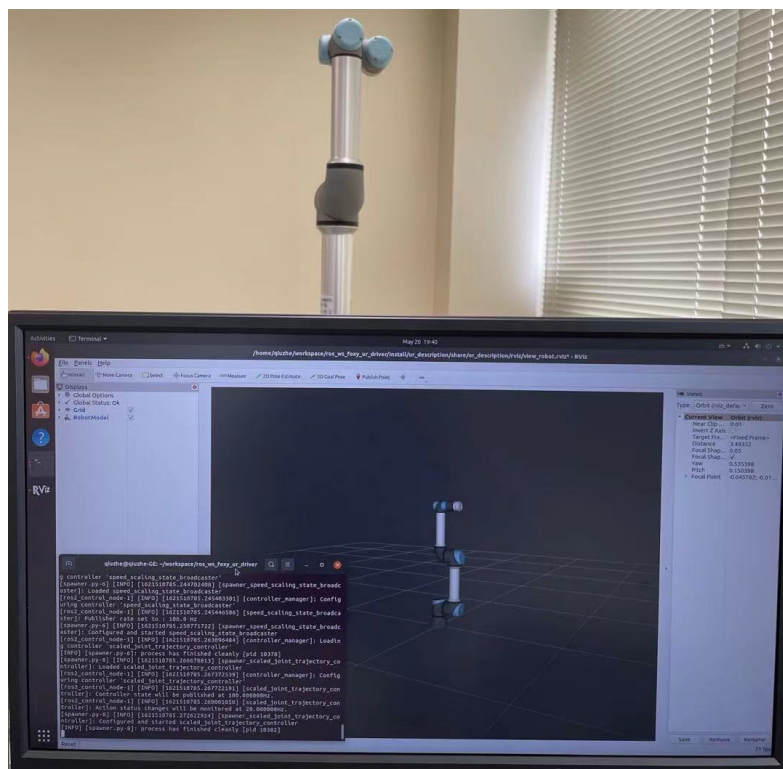
Open Terminal 2:

```
$ export COLCON_WS=~/.workspace/ros_ws_foxy_ur_driver
```

```
$ cd $COLCON_WS
```

```
$ source install/setup.bash
```

```
$ ros2 launch ur_bringup ur_control.launch.py ur_type:=ur5e  
robot_ip:=192.168.20.35 robot_controller:=scaled_joint_trajectory_controller  
launch_rviz:=true
```



As shown in the above figure, the driver is already started.

Step 5:

Load Robot Program: External Control, and start it. Same as Step 5 of Example 3.2

Step 6:

Start the Scaled Joint Trajectory Controller. Remember source the bash file first.

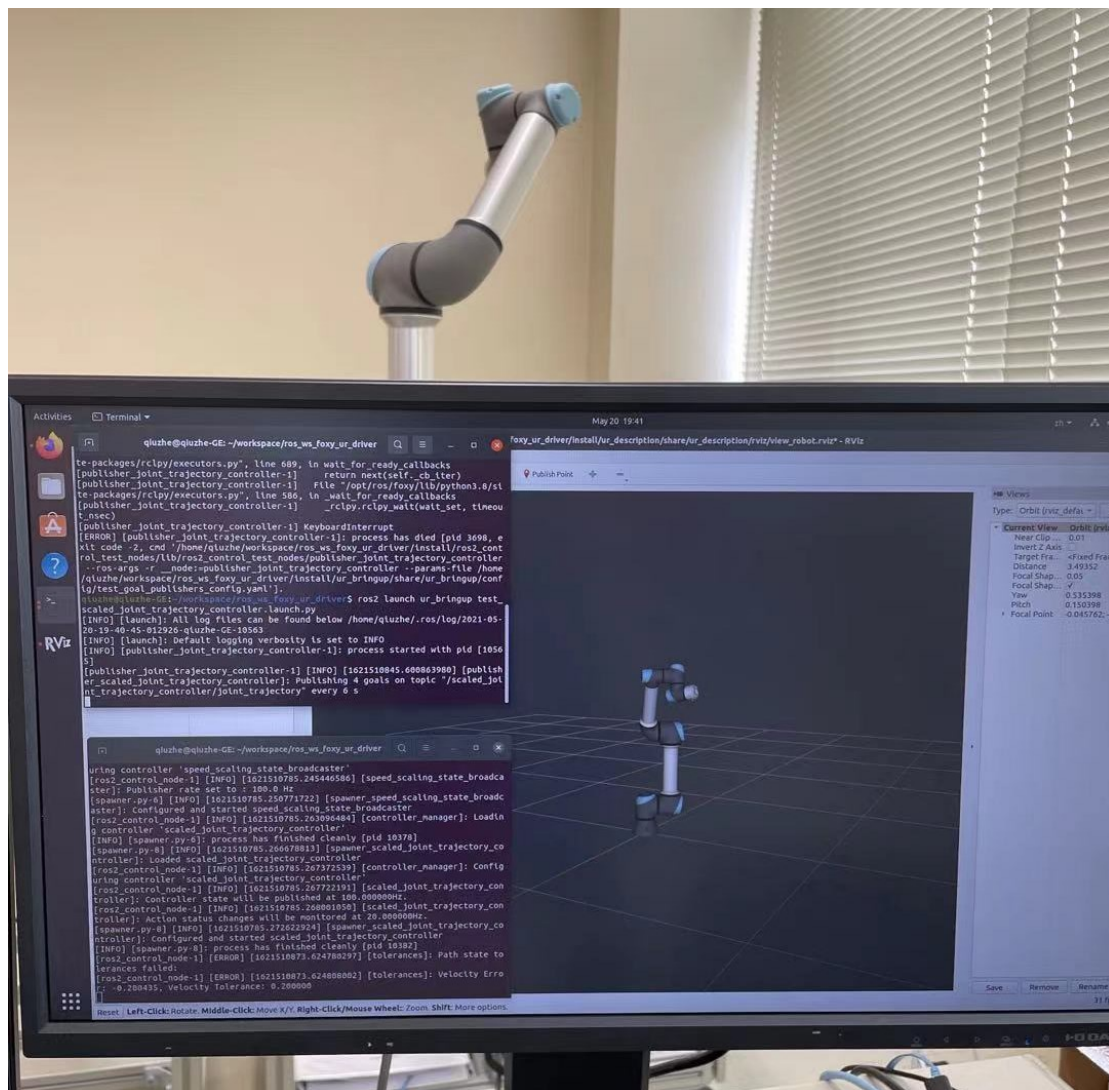
Open Terminal 3:

```
$ export COLCON_WS=~/.workspace/ros_ws_foxy_ur_driver
```

```
$ cd $COLCON_WS
```

```
$ source install/setup.bash
```

```
$ ros2 launch ur_bringup test_scaled_joint_trajectory_controller.launch.py
```



After a few seconds, the robot starts to move.

3.4 Test MoveIt plugin

An ur_5e robot is controlled via MoveIt by using a PC (ROS 2 Foxy with Ubuntu 20.04).

The following steps are recommended:

Step 1:

Same as Step 1 of Example 3.2

Step 2:

Same as Step 2 of Example 3.2

Step 3:

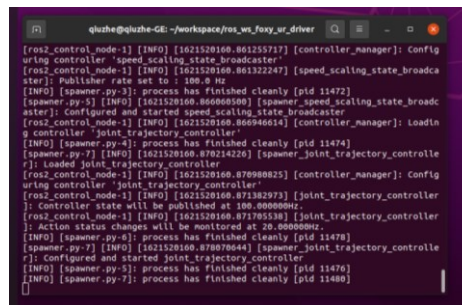
Same as Step 3 of Example 3.2

Step 4:

Start the robot driver. Remember source the bash file first.

Open Terminal 2:

```
$ export COLCON_WS=~/.workspace/ros_ws_foxy_ur_driver
$ cd $COLCON_WS
$ source install/setup.bash
$ ros2 launch ur_bringup ur_control.launch.py ur_type:=ur5e
robot_ip:=192.168.20.35 launch_rviz:=false
```



```
qizhe@qizhe-GE: ~/.workspace/ros_ws_foxy_ur_driver
[ros2_control_node-1] [INFO] [1621520160.861253717] [controller_manager]: Config
uring controller 'speed_scaling_state_broadcaster'
[ros2_control_node-1] [INFO] [1621520160.861322247] [speed_scaling_state_broadc
aster]: Publisher rate set to : 100.0 Hz
[INFO] [spawner.py-3]: process has finished cleanly [pid 11472]
[spawner.py-5] [INFO] [1621520160.860608500] [spawner_speed_scaling_state_broadc
aster]: Configured and started speed_scaling_state_broadcaster
[ros2_control_node-1] [INFO] [1621520160.860940614] [controller_manager]: Loadin
g controller 'joint_trajectory_controller'
[INFO] [spawner.py-4]: process has finished cleanly [pid 11474]
[spawner.py-7] [INFO] [1621520160.870214226] [spawner_joint_trajectory_controlle
r]: Loaded joint_trajectory_controller
[ros2_control_node-1] [INFO] [1621520160.871302973] [joint_trajectory_controller
]: Controller state will be published at 100.000000Hz
[ros2_control_node-1] [INFO] [1621520160.871705530] [joint_trajectory_controller
]: Action status changes will be monitored at 20.000000Hz
[INFO] [spawner.py-6]: process has finished cleanly [pid 11476]
[spawner.py-7] [INFO] [1621520160.878078644] [spawner_joint_trajectory_controlle
r]: Configured and started joint_trajectory_controller
[INFO] [spawner.py-5]: process has finished cleanly [pid 11470]
[INFO] [spawner.py-7]: process has finished cleanly [pid 11480]
```

As shown in the above figure, the driver is successfully started.

Step 5:

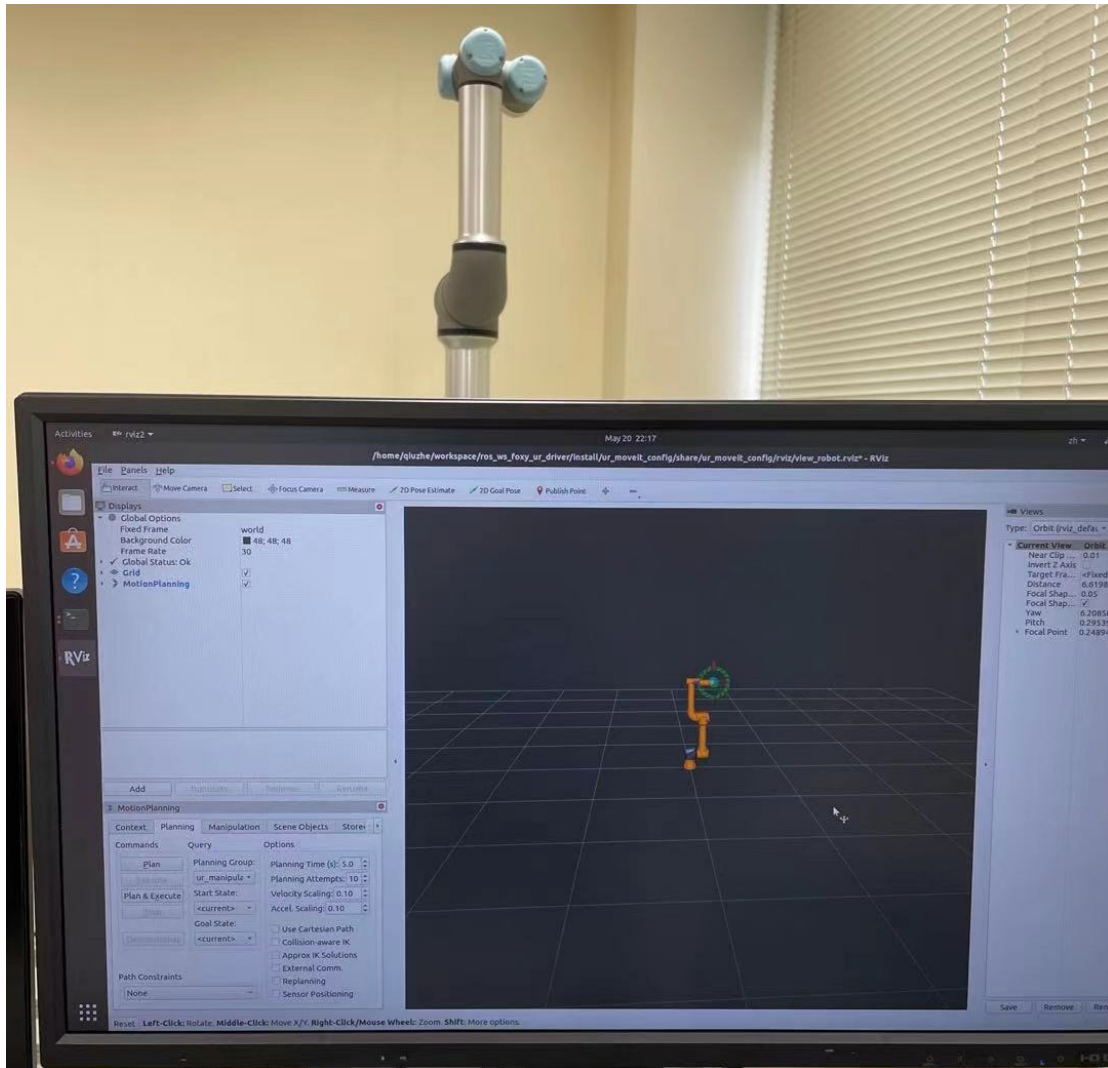
Load Robot Program: External Control, and start it. Same as Step 5 of Example 3.2

Step 6:

Start the MoveIt example. Remember source the bash file first.

Open Terminal 3:

```
$ export COLCON_WS=~/.workspace/ros_ws_foxy_ur_driver
$ cd $COLCON_WS
$ source install/setup.bash
$ ros2 launch ur_bringup ur_moveit.launch.py ur_type:=ur5e
robot_ip:=192.168.20.35 launch_rviz:=true
```



As shown in the above figure, now you can use the MoveIt Plugin in rviz2 to plan and execute trajectories with the robot.