

UR5 control using ROS2

```
sudo apt-get update && sudo apt-get upgrade -y && sudo apt-get dist-upgrade -y
```

```
sudo apt-get install python3-colcon-ros
```

```
sudo apt-get install python3-colcon-bash
```

```
sudo apt update && sudo apt install curl gnupg2 lsb-release
```

```
sudo curl -sSL https://raw.githubusercontent.com/ros/rosdistro/master/ros.key -o /usr/share/keyrings/ros-archive-keyring.gpg
```

```
echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/ros-archive-keyring.gpg] http://packages.ros.org/ros2/ubuntu $(source /etc/os-release && echo $UBUNTU_CODENAME) main" | sudo tee /etc/apt/sources.list.d/ros2.list > /dev/null
```

```
sudo apt update
```

```
sudo apt-get upgrade
```

```
sudo apt install ros-foxy-desktop
```

```
sudo apt install ros-foxy-moveit
```

```
sudo apt install ros-foxy-ros2-control
```

```
sudo apt install ros-foxy-rqt-service-caller
```

```
sudo apt install ros-foxy-warehouse-ros-mongo
```

```
mkdir ur5
```

```
cd ur5
```

```
mkdir ros2
```

```
cd ros2
```

```
mkdir src
```

```
cd src
```

```
git clone --recurse-submodules -j8 https://github.com/ashacs2/ur5_interface.git -b ros2
```

```
git clone --recurse-submodules -j8
```

```
https://github.com/ashacs2/ur5_ros2_interface.git -b ros2
```

```
gedit ~/.bashrc
```

```
source ~/ur5/ros2/install/setup.bash
```

```
cd ur5/ros2/
```

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

```
cd ..
```

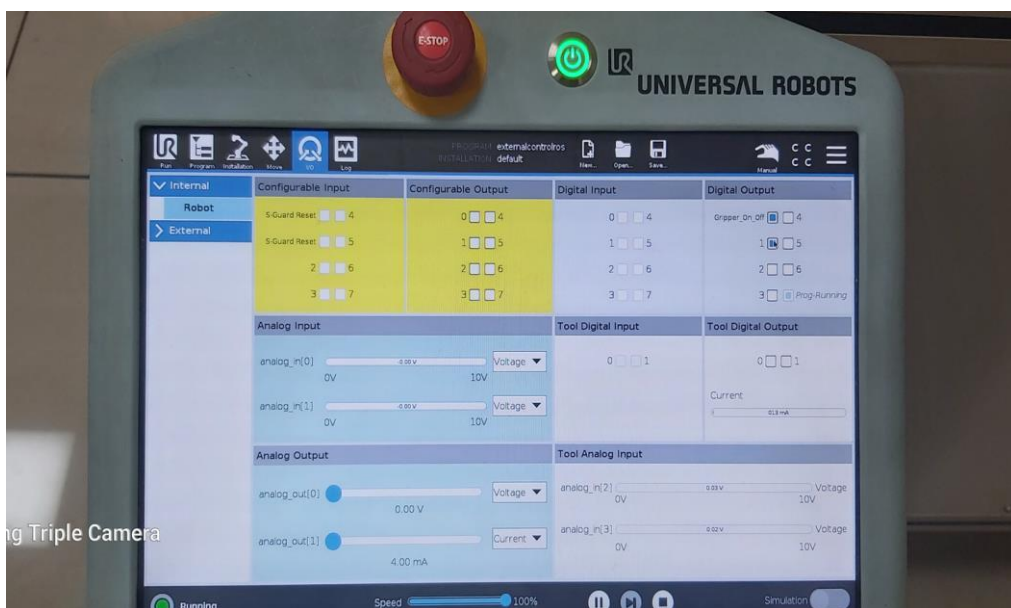
```
colcon build --symlink-install --cmake-args -
```

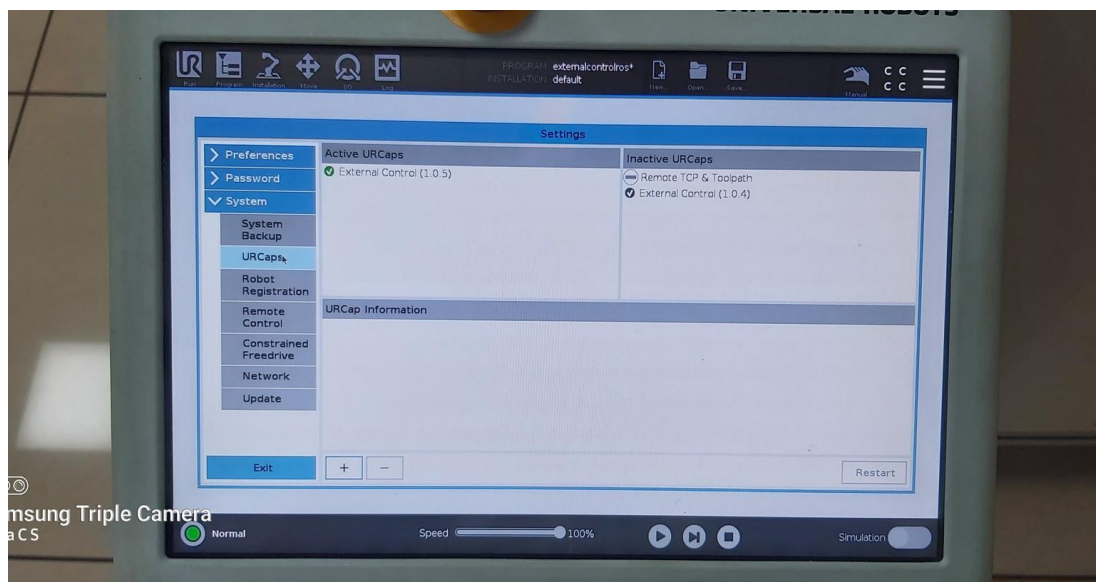
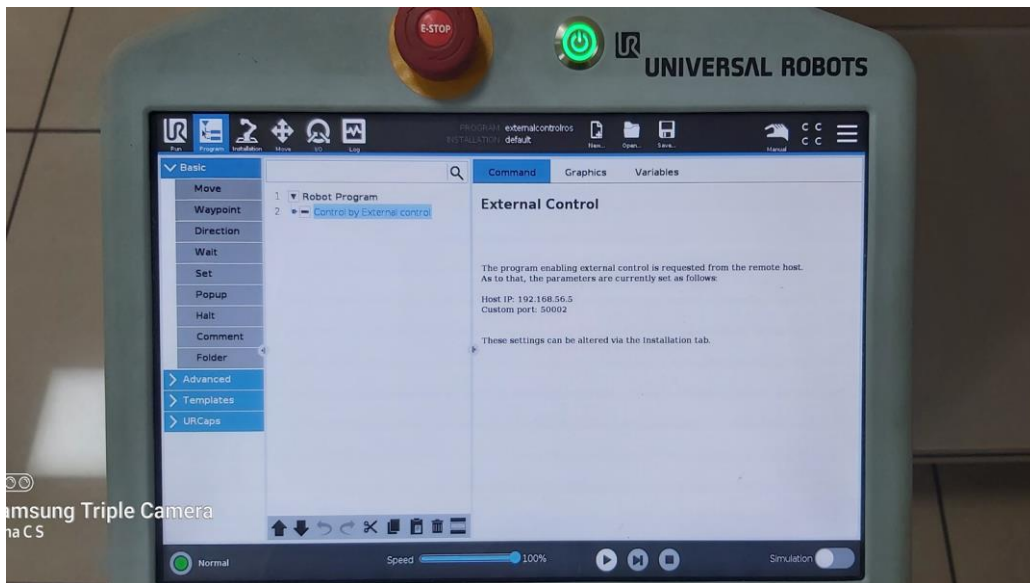
```
DCMAKE_EXPORT_COMPILE_COMMANDS=1
```

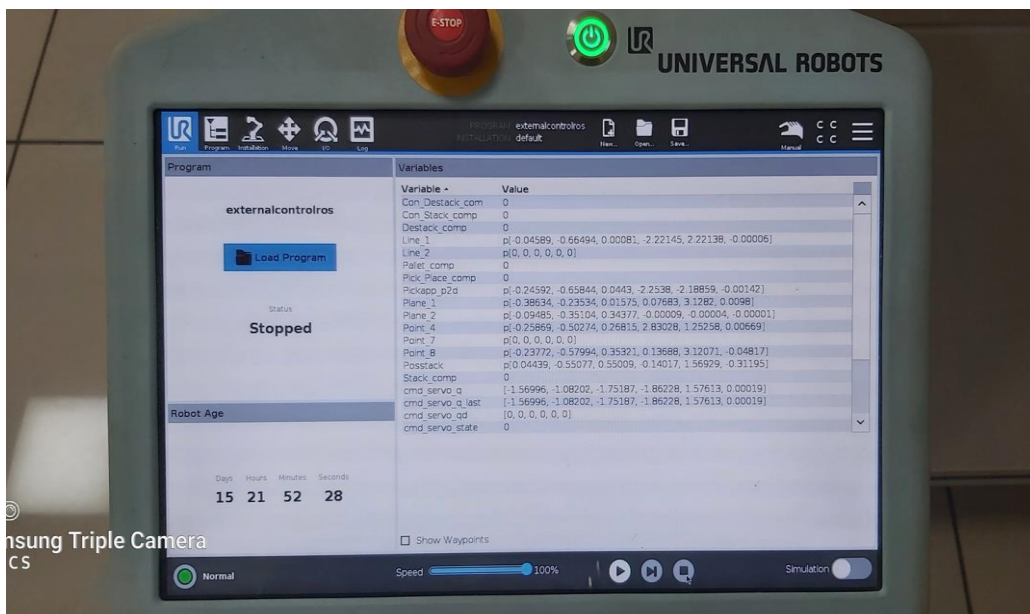
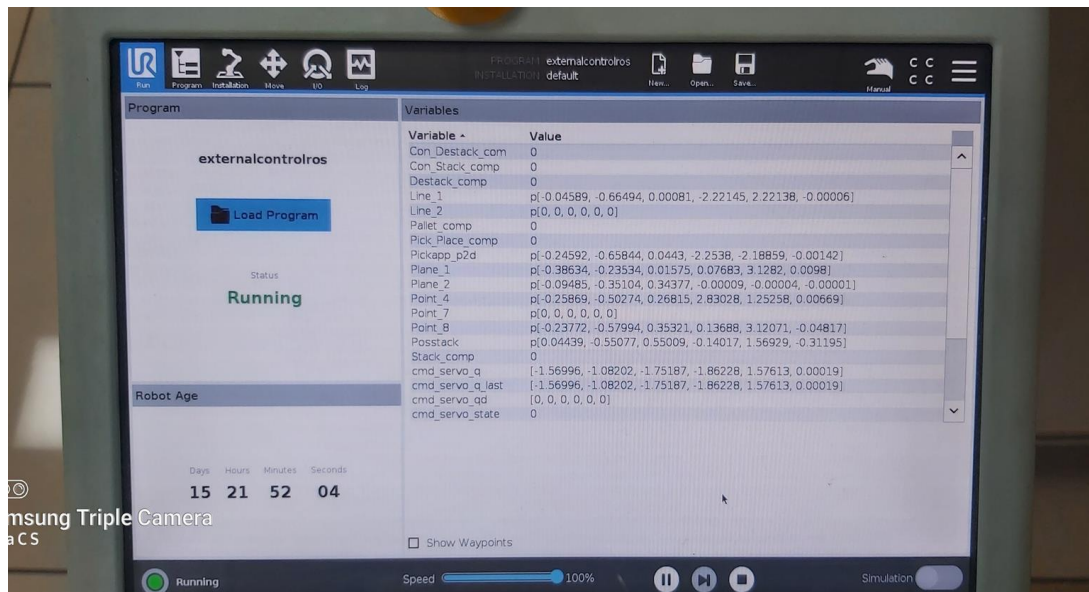
Connect an ethernet cable from UR5 to PC.

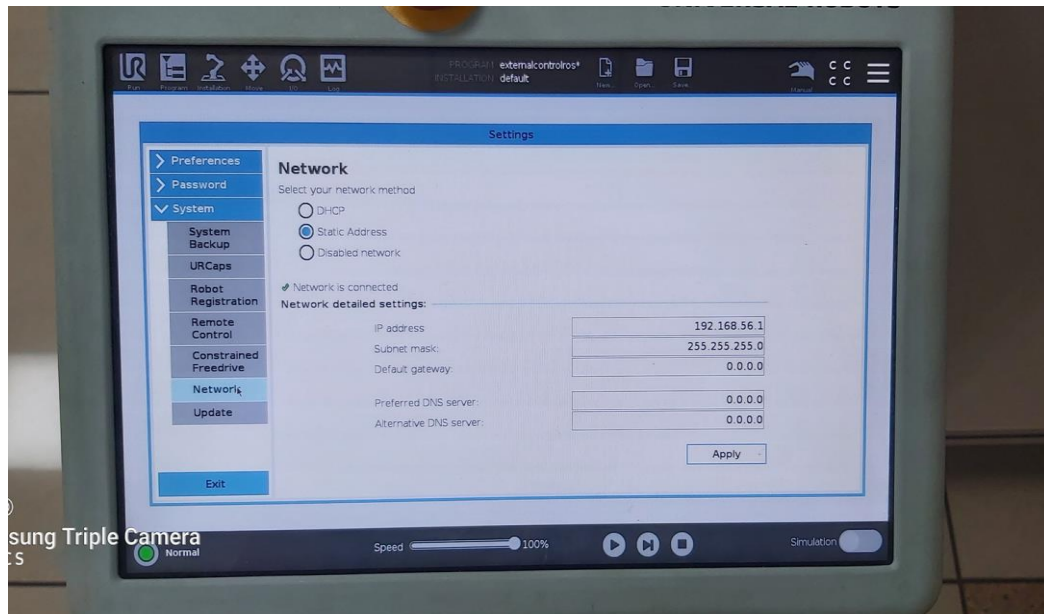


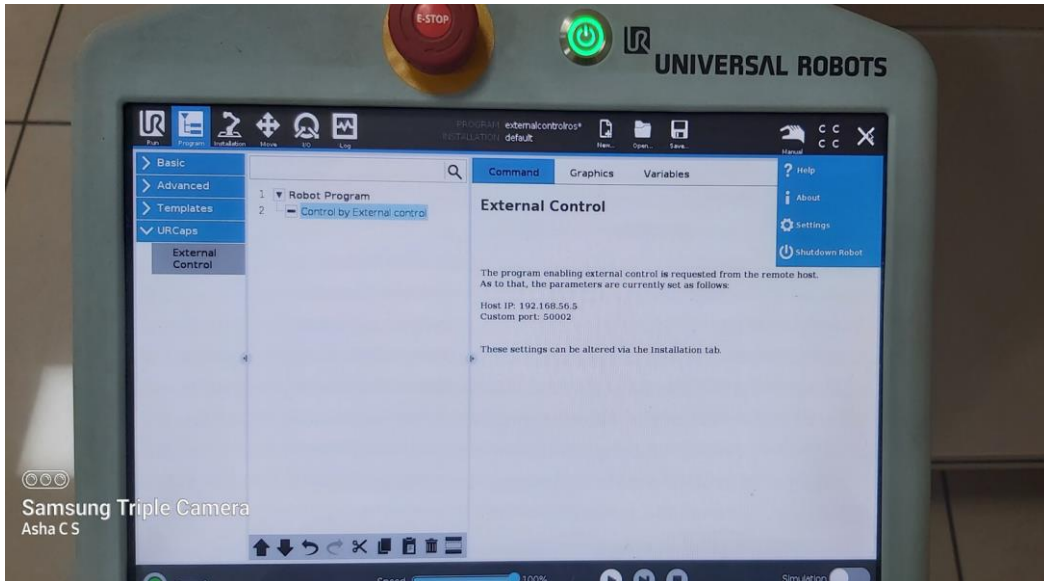
Settings in Teach Pendant:









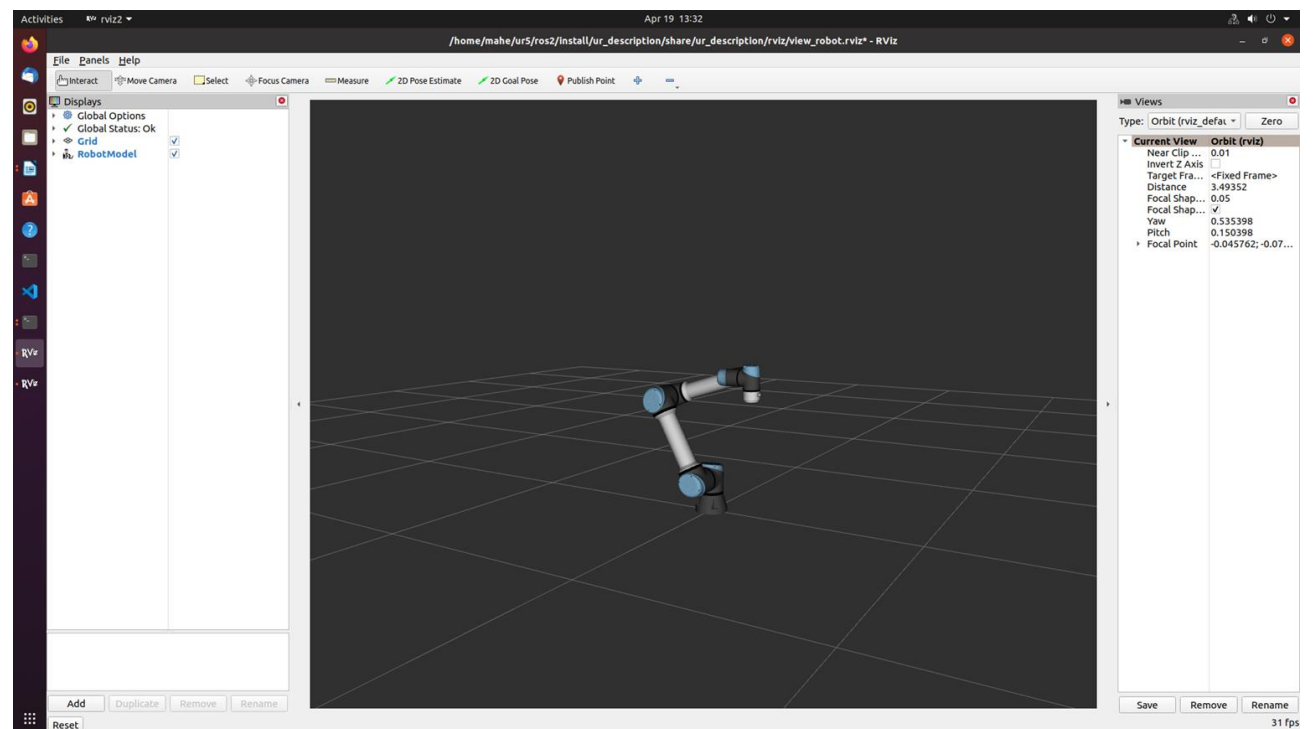


ros2 launch ur_bringup ur5e.launch.py robot_ip:=192.168.56.1

```

Activities  Terminal  Apr 19 13:33
mahe@mahe-HP-Z230-SFF-Workstation: ~
[rviz2-4] at line 133 in /tmp/binarydeb/ros-foxy-tf2-0.13.13/src/buffer_core.cpp
[rviz2-4] Warning: Invalid frame ID "wrist_2_link" passed to canTransform argument source_frame - frame does not exist
[rviz2-4] at line 133 in /tmp/binarydeb/ros-foxy-tf2-0.13.13/src/buffer_core.cpp
[ros2_control_node-1] [INFO] [1650355016.671245041] [controller_manager]: Loading controller 'force_torque_sensor_broadcaster'
[spawner.py-8] [INFO] [1650355016.676802447] [spawner_force_torque_sensor_broadcaster]: Loaded force_torque_sensor_broadcaster
[ros2_control_node-1] [INFO] [1650355016.677654494] [controller_manager]: Configuring controller 'force_torque_sensor_broadcaster'
[spawner.py-8] [INFO] [1650355016.68437788] [spawner_force_torque_sensor_broadcaster]: Configured and started force_torque_sensor_broadcaster
[rviz2-4] Warning: Invalid frame ID "forearm_link" passed to canTransform argument source_frame - frame does not exist
[rviz2-4] at line 133 in /tmp/binarydeb/ros-foxy-tf2-0.13.13/src/buffer_core.cpp
[rviz2-4] Warning: Invalid frame ID "shoulder_link" passed to canTransform argument source_frame - frame does not exist
[rviz2-4] at line 133 in /tmp/binarydeb/ros-foxy-tf2-0.13.13/src/buffer_core.cpp
[rviz2-4] Warning: Invalid frame ID "upper_arm_link" passed to canTransform argument source_frame - frame does not exist
[rviz2-4] at line 133 in /tmp/binarydeb/ros-foxy-tf2-0.13.13/src/buffer_core.cpp
[rviz2-4] Warning: Invalid frame ID "wrist_1_link" passed to canTransform argument source_frame - frame does not exist
[rviz2-4] at line 133 in /tmp/binarydeb/ros-foxy-tf2-0.13.13/src/buffer_core.cpp
[rviz2-4] Warning: Invalid frame ID "wrist_2_link" passed to canTransform argument source_frame - frame does not exist
[rviz2-4] at line 133 in /tmp/binarydeb/ros-foxy-tf2-0.13.13/src/buffer_core.cpp
[rviz2-4] Warning: Invalid frame ID "forearm_link" passed to canTransform argument source_frame - frame does not exist
[rviz2-4] at line 133 in /tmp/binarydeb/ros-foxy-tf2-0.13.13/src/buffer_core.cpp
[rviz2-4] Warning: Invalid frame ID "shoulder_link" passed to canTransform argument source_frame - frame does not exist
[rviz2-4] at line 133 in /tmp/binarydeb/ros-foxy-tf2-0.13.13/src/buffer_core.cpp
[rviz2-4] Warning: Invalid frame ID "upper_arm_link" passed to canTransform argument source_frame - frame does not exist
[rviz2-4] at line 133 in /tmp/binarydeb/ros-foxy-tf2-0.13.13/src/buffer_core.cpp
[rviz2-4] Warning: Invalid frame ID "wrist_1_link" passed to canTransform argument source_frame - frame does not exist
[rviz2-4] at line 133 in /tmp/binarydeb/ros-foxy-tf2-0.13.13/src/buffer_core.cpp
[rviz2-4] Warning: Invalid frame ID "wrist_2_link" passed to canTransform argument source_frame - frame does not exist
[rviz2-4] at line 133 in /tmp/binarydeb/ros-foxy-tf2-0.13.13/src/buffer_core.cpp
[ros2_control_node-1] [INFO] [1650355016.73610796] [controller_manager]: Loading controller 'joint_state_broadcaster'
[spawner.py-5] [INFO] [1650355016.744699883] [spawner_joint_state_broadcaster]: Loaded joint_state_broadcaster
[ros2_control_node-1] [INFO] [1650355016.745931924] [controller_manager]: Configuring controller 'joint_state_broadcaster'
[rviz2-4] Warning: Invalid frame ID "forearm_link" passed to canTransform argument source_frame - frame does not exist
[rviz2-4] at line 133 in /tmp/binarydeb/ros-foxy-tf2-0.13.13/src/buffer_core.cpp
[rviz2-4] Warning: Invalid frame ID "shoulder_link" passed to canTransform argument source_frame - frame does not exist
[rviz2-4] at line 133 in /tmp/binarydeb/ros-foxy-tf2-0.13.13/src/buffer_core.cpp
[rviz2-4] Warning: Invalid frame ID "upper_arm_link" passed to canTransform argument source_frame - frame does not exist
[rviz2-4] at line 133 in /tmp/binarydeb/ros-foxy-tf2-0.13.13/src/buffer_core.cpp
[rviz2-4] Warning: Invalid frame ID "wrist_1_link" passed to canTransform argument source_frame - frame does not exist
[rviz2-4] at line 133 in /tmp/binarydeb/ros-foxy-tf2-0.13.13/src/buffer_core.cpp
[rviz2-4] Warning: Invalid frame ID "wrist_2_link" passed to canTransform argument source_frame - frame does not exist
[rviz2-4] at line 133 in /tmp/binarydeb/ros-foxy-tf2-0.13.13/src/buffer_core.cpp
[spawner.py-5] [INFO] [1650355016.755802593] [spawner_joint_state_broadcaster]: Configured and started joint_state_broadcaster
[INFO] [spawner.py-5]: process has finished cleanly [pid 125550]
[ros2_control_node-1] [INFO] [1650355016.830802928] [controller_manager]: Loading controller 'io_and_status_controller'
[spawner.py-6] [INFO] [1650355016.847411038] [spawner_io_and_status_controller]: Loaded io_and_status_controller
[ros2_control_node-1] [INFO] [1650355016.848411666] [controller_manager]: Configuring controller 'io_and_status_controller'
[spawner.py-6] [INFO] [1650355016.855437181] [spawner_io_and_status_controller]: Configured and started io_and_status_controller
[INFO] [spawner.py-6]: process has finished cleanly [pid 125579]
[ros2_control_node-1] [INFO] [1650355107.009735335] [joint_trajectory_controller]: Received new action goal
[ros2_control_node-1] [INFO] [1650355107.009802997] [joint_trajectory_controller]: Accepted new action goal
[ros2_control_node-1] [INFO] [1650355108.850877485] [joint_trajectory_controller]: Goal reached, success!
[ros2_control_node-1] [INFO] [1650355115.221215176] [joint_trajectory_controller]: Received new action goal
[ros2_control_node-1] [INFO] [1650355115.221260921] [joint_trajectory_controller]: Accepted new action goal
[ros2_control_node-1] [INFO] [1650355116.791774396] [joint_trajectory_controller]: Goal reached, success!

```

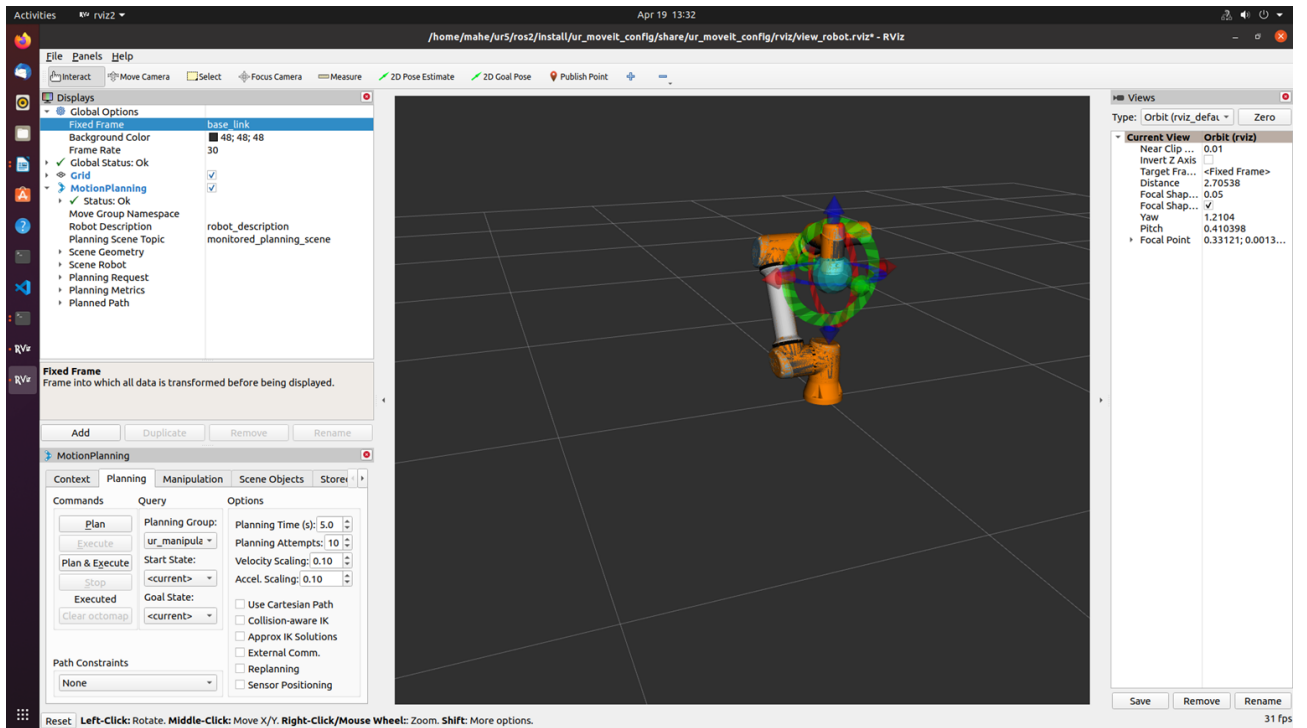


Set in manual mode (1234)
 Open externalcontrolros program
 Run the program

ros2 launch ur_bringup ur_moveit.launch.py ur_type:=ur5e
 robot_ip:=192.168.56.1

```

[move_group-1] [INFO] [1650355115.195789282] [ompl]: /tmp/binarydeb/ros-foxy-ompl-1.5.0/src/ompl/geometric/planners/rrt/src/RRTConnect.cpp:220 - ur_manipulator/ur_manipulator: Starting planning with 1 state
[move_group-1] [INFO] [1650355115.196170482] [ompl]: /tmp/binarydeb/ros-foxy-ompl-1.5.0/src/ompl/geometric/planners/rrt/src/RRTConnect.cpp:354 - ur_manipulator/ur_manipulator: Created 4 states (2 start + 2
[move_group-1] [INFO] [1650355115.196482472] [ompl]: /tmp/binarydeb/ros-foxy-ompl-1.5.0/src/ompl/geometric/planners/rrt/src/RRTConnect.cpp:220 - ur_manipulator/ur_manipulator: Starting planning with 1 state
[move_group-1] [INFO] [1650355115.197536559] [ompl]: /tmp/binarydeb/ros-foxy-ompl-1.5.0/src/ompl/geometric/planners/rrt/src/RRTConnect.cpp:354 - ur_manipulator/ur_manipulator: Created 4 states (2 start + 2
[move_group-1] [INFO] [1650355115.198066550] [ompl]: /tmp/binarydeb/ros-foxy-ompl-1.5.0/src/ompl/geometric/planners/rrt/src/RRTConnect.cpp:354 - ur_manipulator/ur_manipulator: Created 4 states (2 start + 2
[move_group-1] [INFO] [1650355115.206111188] [ompl]: /tmp/binarydeb/ros-foxy-ompl-1.5.0/src/ompl/geometric/planners/rrt/src/RRTConnect.cpp:354 - ur_manipulator/ur_manipulator: Created 4 states (2 start + 2
[move_group-1] [INFO] [1650355115.206193765] [ompl]: /tmp/binarydeb/ros-foxy-ompl-1.5.0/src/ompl/tools/multiplan/src/ParallelPlan.cpp:135 - ParallelPlan::solve(): Solution found by one or more threads in 0.
011187 seconds
[move_group-1] [INFO] [1650355115.206391710] [ompl]: /tmp/binarydeb/ros-foxy-ompl-1.5.0/src/ompl/geometric/planners/rrt/src/RRTConnect.cpp:220 - ur_manipulator/ur_manipulator: Starting planning with 1 state
[move_group-1] [INFO] [1650355115.206691074] [ompl]: /tmp/binarydeb/ros-foxy-ompl-1.5.0/src/ompl/geometric/planners/rrt/src/RRTConnect.cpp:220 - ur_manipulator/ur_manipulator: Starting planning with 1 state
[move_group-1] [INFO] [1650355115.207103525] [ompl]: /tmp/binarydeb/ros-foxy-ompl-1.5.0/src/ompl/geometric/planners/rrt/src/RRTConnect.cpp:220 - ur_manipulator/ur_manipulator: Starting planning with 1 state
[move_group-1] [INFO] [1650355115.207421250] [ompl]: /tmp/binarydeb/ros-foxy-ompl-1.5.0/src/ompl/geometric/planners/rrt/src/RRTConnect.cpp:354 - ur_manipulator/ur_manipulator: Created 4 states (2 start + 2
[move_group-1] [INFO] [1650355115.207437607] [ompl]: /tmp/binarydeb/ros-foxy-ompl-1.5.0/src/ompl/geometric/planners/rrt/src/RRTConnect.cpp:354 - ur_manipulator/ur_manipulator: Created 5 states (3 start + 2
[move_group-1] [INFO] [1650355115.207531904] [ompl]: /tmp/binarydeb/ros-foxy-ompl-1.5.0/src/ompl/geometric/planners/rrt/src/RRTConnect.cpp:220 - ur_manipulator/ur_manipulator: Starting planning with 1 state
[move_group-1] [INFO] [1650355115.208848779] [ompl]: /tmp/binarydeb/ros-foxy-ompl-1.5.0/src/ompl/geometric/planners/rrt/src/RRTConnect.cpp:354 - ur_manipulator/ur_manipulator: Created 5 states (2 start + 3
[move_group-1] [INFO] [1650355115.209053743] [ompl]: /tmp/binarydeb/ros-foxy-ompl-1.5.0/src/ompl/geometric/planners/rrt/src/RRTConnect.cpp:354 - ur_manipulator/ur_manipulator: Created 4 states (2 start + 2
[move_group-1] [INFO] [1650355115.210213560] [ompl]: /tmp/binarydeb/ros-foxy-ompl-1.5.0/src/ompl/tools/multiplan/src/ParallelPlan.cpp:135 - ParallelPlan::solve(): Solution found by one or more threads in 0.
003951 seconds
[move_group-1] [INFO] [1650355115.210340921] [ompl]: /tmp/binarydeb/ros-foxy-ompl-1.5.0/src/ompl/geometric/planners/rrt/src/RRTConnect.cpp:220 - ur_manipulator/ur_manipulator: Starting planning with 1 state
[move_group-1] [INFO] [1650355115.211799692] [ompl]: /tmp/binarydeb/ros-foxy-ompl-1.5.0/src/ompl/geometric/planners/rrt/src/RRTConnect.cpp:354 - ur_manipulator/ur_manipulator: Created 5 states (3 start + 2
[move_group-1] [INFO] [1650355115.211947862] [ompl]: /tmp/binarydeb/ros-foxy-ompl-1.5.0/src/ompl/geometric/planners/rrt/src/RRTConnect.cpp:220 - ur_manipulator/ur_manipulator: Starting planning with 1 state
[move_group-1] [INFO] [1650355116.000322244] [ompl]: /tmp/binarydeb/ros-foxy-ompl-1.5.0/src/ompl/geometric/planners/rrt/src/RRTConnect.cpp:354 - ur_manipulator/ur_manipulator: Created 5 states (3 start + 2
[move_group-1] [INFO] [1650355116.213072402] [ompl]: /tmp/binarydeb/ros-foxy-ompl-1.5.0/src/ompl/tools/multiplan/src/ParallelPlan.cpp:135 - ParallelPlan::solve(): Solution found by one or more threads in 0.
002895 seconds
[move_group-1] [INFO] [1650355116.216738628] [ompl]: /tmp/binarydeb/ros-foxy-ompl-1.5.0/src/ompl/geometric/planners/rrt/src/RRTConnect.cpp:179 - SimpleSetup: Path simplification took 0.003629 seconds and changed from 3
to 2 states
[move_group-1] [INFO] [1650355116.218459227] [moveit_ros_trajectory_execution_manager]: Validating trajectory with allowed_start_tolerance 0.01
[move_group-1] [INFO] [1650355116.220191202] [moveit_ros_trajectory_execution_manager]: Starting trajectory execution ...
[move_group-1] [INFO] [1650355116.220241858] [moveit_simple_controller_manager.follow_joint_trajectory_controller_handle]: sending trajectory to joint_trajectory_controller
[move_group-1] [INFO] [1650355116.221035758] [moveit_simple_controller_manager.follow_joint_trajectory_controller_handle]: joint_trajectory_controller started execution
[move_group-1] [INFO] [1650355116.221852729] [moveit_simple_controller_manager.follow_joint_trajectory_controller_handle]: joint_trajectory_controller finished execution
[move_group-1] [INFO] [1650355116.493672144] [move_group_interface]: Plan and Execute request accepted
[move_group-1] [INFO] [1650355116.791853548] [moveit_simple_controller_manager.follow_joint_trajectory_controller_handle]: Controller joint_trajectory_controller successfully finished
[move_group-1] [INFO] [1650355116.799255061] [moveit_ros_trajectory_execution_manager]: Completed trajectory execution with status SUCCEEDED ...
[move_group-1] [INFO] [1650355116.800322244] [moveit_move_group_default_capabilities.move_action_capability]: Solution was found and executed.
[move_group-1] [INFO] [1650355117.095315225] [move_group_interface]: Plan and Execute request complete!
  
```



type >>
 rqt
 plugins→service caller
 select io_and_status_controller/set_io
 set the pin as 0 for gripper
 state as 1.0 to set the gripper off
 state as 0.0 to set the gripper off

Activities rqt May 4 14:03 Default - rqt

File Plugins Running Perspectives Help

Service Caller (2)

Service /io_and_status_controller/set_io

Request

Topic	Type	Expression
/io_and_status_controller/set_io	ur_msgs/srv/SetIO	
fun	int8	1
pin	int8	1
state	float	1.0

Response

Field	Type	Value
/	ur_msgs/srv/SetIO.Response	
success	boolean	True

Activities rqt May 4 14:01 Default - rqt

File Plugins Running Perspectives Help

Service Caller (2)

Service /io_and_status_controller/set_io

Request

Topic	Type	Expression
/io_and_status_controller/set_io	ur_msgs/srv/SetIO	
fun	int8	1
pin	int8	0
state	float	1.0

Response

Field	Type	Value
/	ur_msgs/srv/SetIO.Response	
success	boolean	True

```

Activities Terminal May 4 14:01
mahe@mahe-HP-Z230-SFF-Workstation: ~/urs
mahe@mahe-HP-Z230-SFF-Workstation: ~/urs
mahe@mahe-HP-Z230-SFF-Workstation: ~/urs$ ros2 service type /io_and_status_controller/set_io
ur_msgs/srv/SetIO
mahe@mahe-HP-Z230-SFF-Workstation: ~/urs$ ros2 interface show ur_msgs/srv/SetIO
# Service allows setting digital (DIO) and analog (AIO) IO, as well as flags
# and configuring tool voltage.
#
# This service has three request fields (see below).
#
# When setting DIO or AIO pins to new values:
#
# fun The function to perform
# pin Numeric ID of the pin to set
# state Desired pin state (signal level for analog or STATE_ON|OFF for DIO and flags)
#
# When configuring tool voltage:
#
# fun Set to FUN_SET_TOOL_VOLTAGE
# pin Ignored
# state Desired tool voltage (use STATE_TOOL_VOLTAGE constants)
#
# constants
#
# valid function values
#
# Note: 'fun' is short for 'function' (ie: the function the service should perform).
int8 FUN_SET_DIGITAL_OUT = 1
int8 FUN_SET_FLAG = 2
int8 FUN_SET_ANALOG_OUT = 3
int8 FUN_SET_TOOL_VOLTAGE = 4
#
# valid values for 'state' when setting digital IO or flags
int8 STATE_OFF = 0
int8 STATE_ON = 1
#
# valid 'state' values when setting tool voltage
int8 STATE_TOOL_VOLTAGE_0V = 0
int8 STATE_TOOL_VOLTAGE_12V = 12
int8 STATE_TOOL_VOLTAGE_24V = 24
#
# request fields
int8 fun
int8 pin
float32 state
---
bool success
mahe@mahe-HP-Z230-SFF-Workstation: ~/urs$
mahe@mahe-HP-Z230-SFF-Workstation: ~/urs$ ros2 run rqt_service_caller rqt_service_caller
mahe@mahe-HP-Z230-SFF-Workstation: ~/urs$ rqt

```

Edit test_goal_publishers_config file as

publisher_scaled_joint_trajectory_controller:

ros__parameters:

controller_name: "scaled_joint_trajectory_controller"

wait_sec_between_publish: 6

goal_names: ["pos1", "pos2", "pos3", "pos4"]

pos1: [-1.76, -2.129, -1.117, -1.483, 1.57, 0.1745]

pos2: [-1.76, -2.3911, -1.448, -0.8726, 1.57, 0.17]

pos3: [-1.76, -2.129, -1.117, -1.483, 1.57, 0.1745]

pos4: [-1.76, -2.3911, -1.448, -0.8726, 1.57, 0.17]

joints:

- shoulder_pan_joint
- shoulder_lift_joint
- elbow_joint
- wrist_1_joint
- wrist_2_joint
- wrist_3_joint

```
check_starting_point: true
starting_point_limits:
  shoulder_pan_joint: [-3.14,3.14]
  shoulder_lift_joint: [-3.14,3.14]
  elbow_joint: [-3.14,3.14]
  wrist_1_joint: [-3.14,3.14]
  wrist_2_joint: [-3.14,3.14]
  wrist_3_joint: [-3.14,3.14]
```

```
publisher_joint_trajectory_controller:
  ros__parameters:
```

```
    controller_name: "joint_trajectory_controller"
    wait_sec_between_publish: 6
```

```
    goal_names: ["pos1", "pos2", "pos3", "pos4"]
    pos1: [-1.76, -2.129, -1.117, -1.483, 1.57, 0.1745]
    pos2: [-1.76, -2.3911, -1.448, -0.8726, 1.57, 0.17]
    pos3: [-1.76, -2.129, -1.117, -1.483, 1.57, 0.1745]
    pos4: [-1.76, -2.3911, -1.448, -0.8726, 1.57, 0.17]
```

```
joints:
  - shoulder_pan_joint
  - shoulder_lift_joint
  - elbow_joint
  - wrist_1_joint
  - wrist_2_joint
  - wrist_3_joint
```

```
check_starting_point: true
starting_point_limits:
  shoulder_pan_joint: [-3.14,3.14]
  shoulder_lift_joint: [-3.14,3.14]
  elbow_joint: [-3.14,3.14]
  wrist_1_joint: [-3.14,3.14]
  wrist_2_joint: [-3.14,3.14]
  wrist_3_joint: [-3.14,3.14]
```

```
cd ur5
colcon build --packages-select ur_bringup
ros2 launch ur_bringup test_joint_trajectory_controller.launch.py
```

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 a 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

3.5 Modified ROS 2 package of scaled-/joint trajectory controller

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) In-

stallation: [https://docs.ros.org/en/foxy/Installation/Ubuntu-Development-](https://docs.ros.org/en/foxy/Installation/Ubuntu-Development-Setup.html)

[Setup.html](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 re-

quired. 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.r
```

```
epos
```

```
$ 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 a 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:

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 Controlprogram 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.

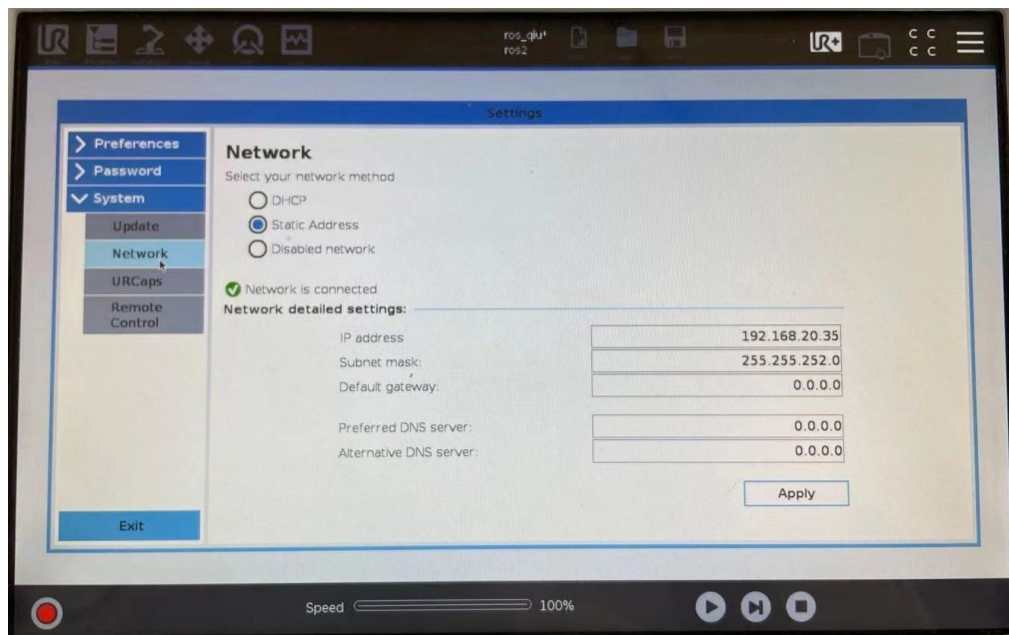
3. Examples

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.

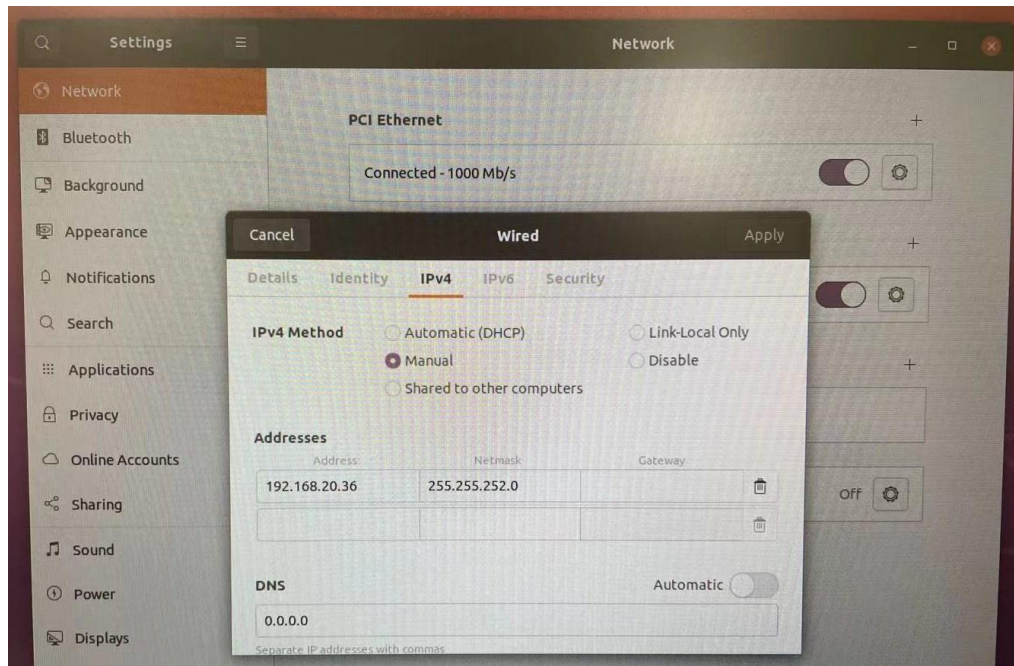
3.1.1 Set IP address of the robot

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

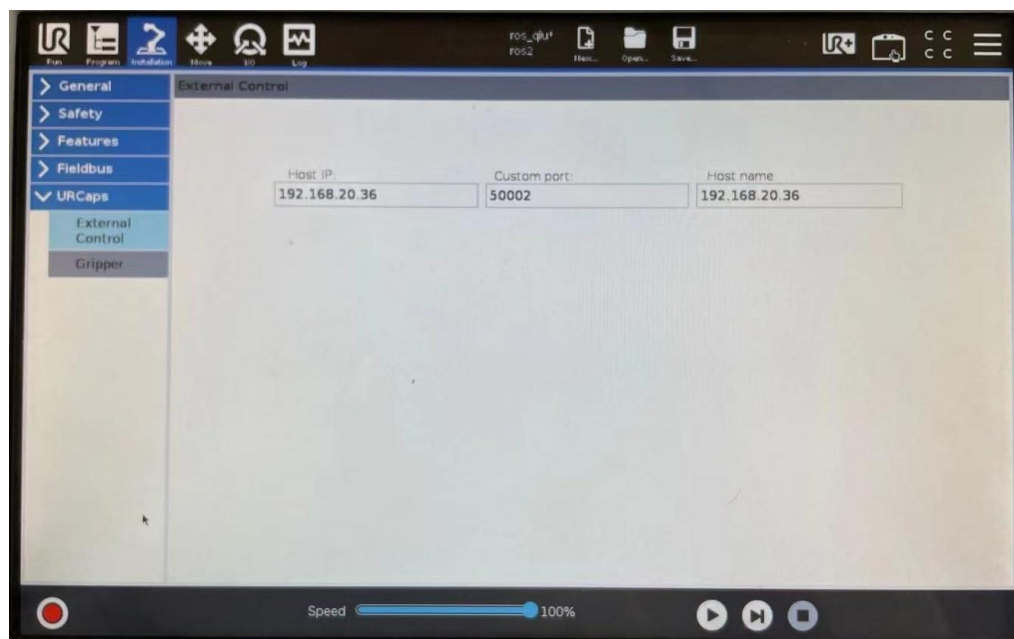


3.1.2 Set IP address of the control PC

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

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

The following steps are recom-

mended: Step 1:

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

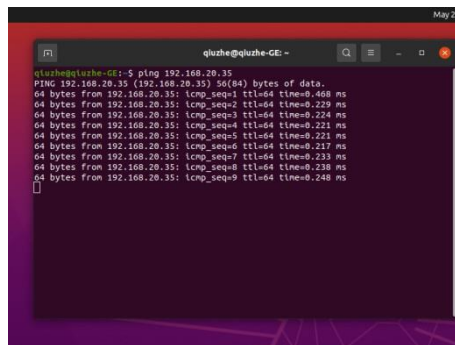
The IP address of the ur_5e robot is set as:

192.168.20.35. The IP address of the PC is set as:

192.168.20.36.

Open Terminal 1:

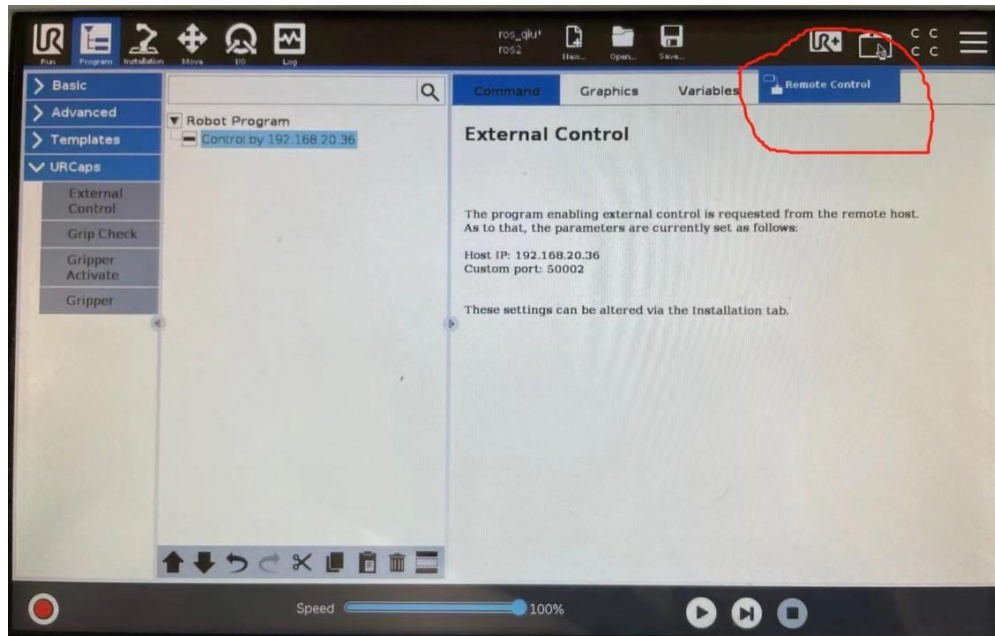
\$ ping 192.168.20.35



We can see the robot and the PC are successfully con-

nected. Step 2:

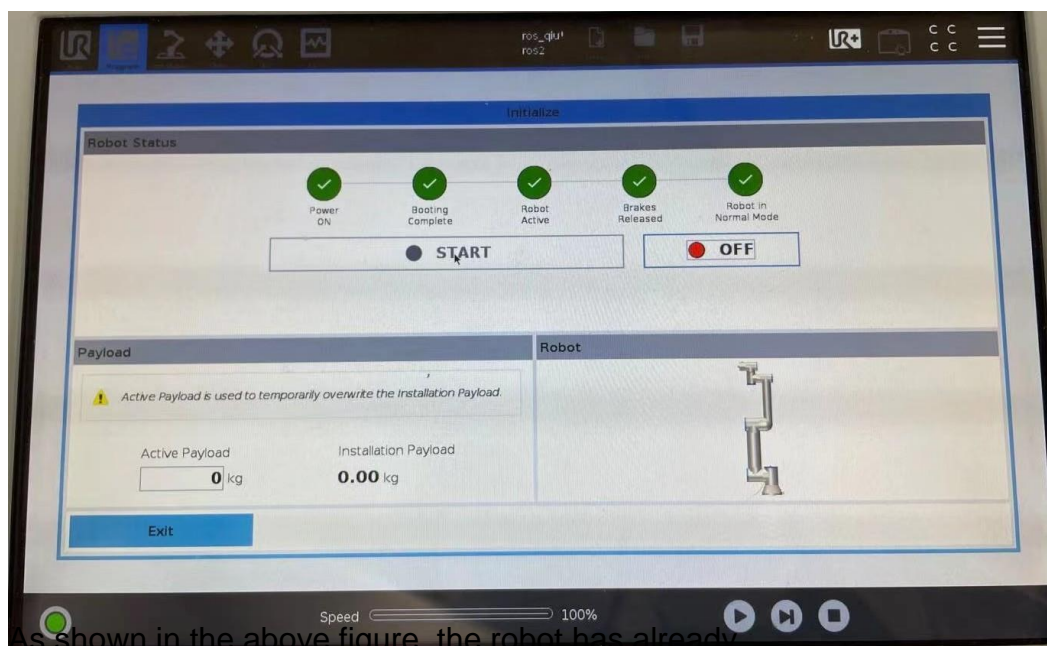
Set the robot control mode to the **local control mode** by using the teach-pendant.



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
```

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

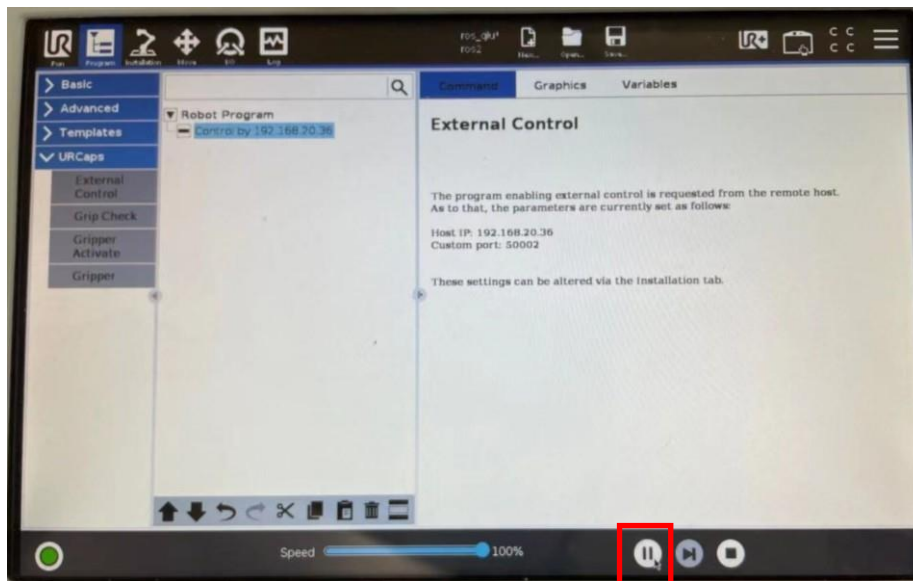
\$ ros2 launch ur_bringup ur_control.launch.py
ur_type:=ur5e robot_ip:=192.168.20.35 launch_rviz:=true



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:

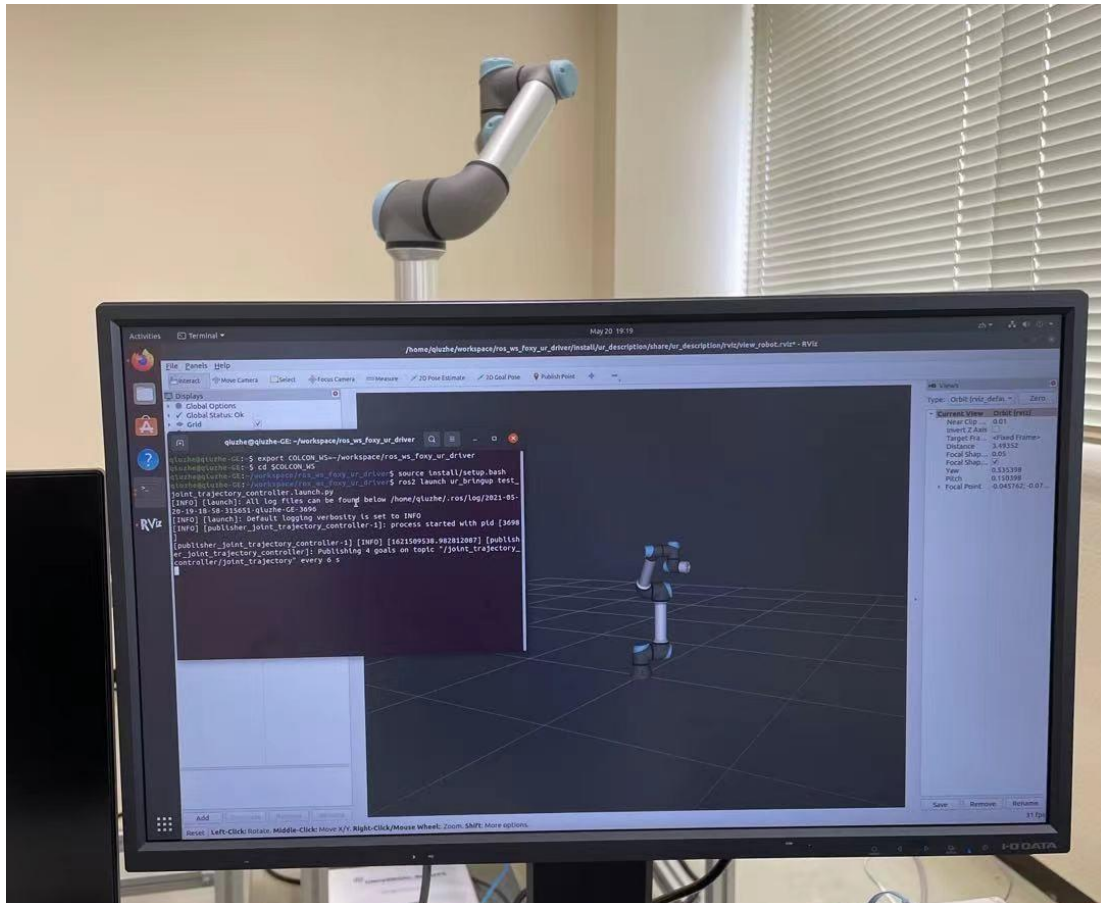
Start the Joint Trajectory Controller. Remember source the bash file first. Open Terminal 3:

```
$ 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

A 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 recom-

mended: 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:

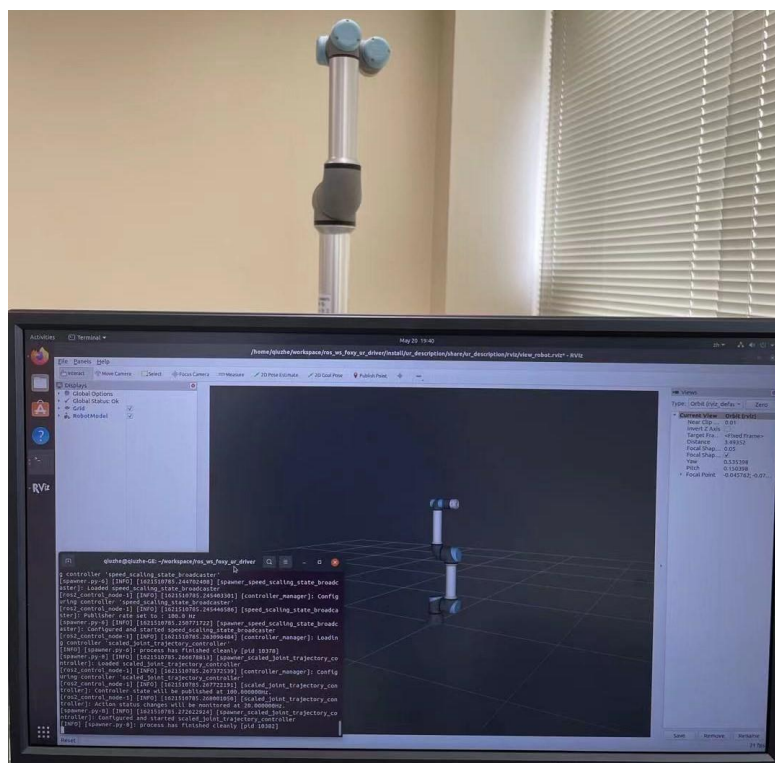
```
$ export COLCON_WS=~/.workspace/ros_ws_foxy_ur_driver
```

```
$ 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:

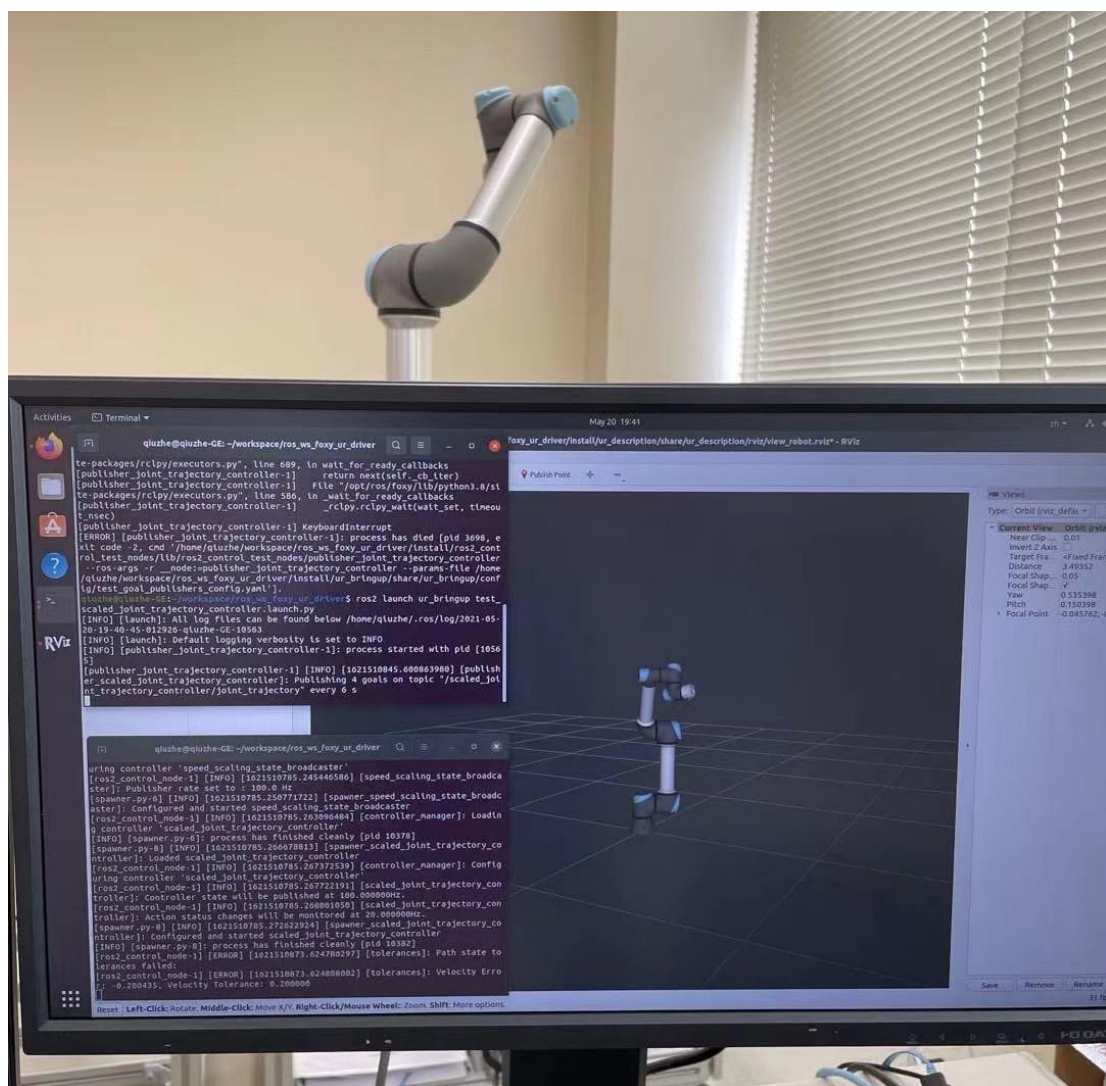
Start the Scaled Joint Trajectory Controller. Remember source the bash file first. Open Terminal 3:

```
$ 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

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

The following steps are recom-

mended: 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:

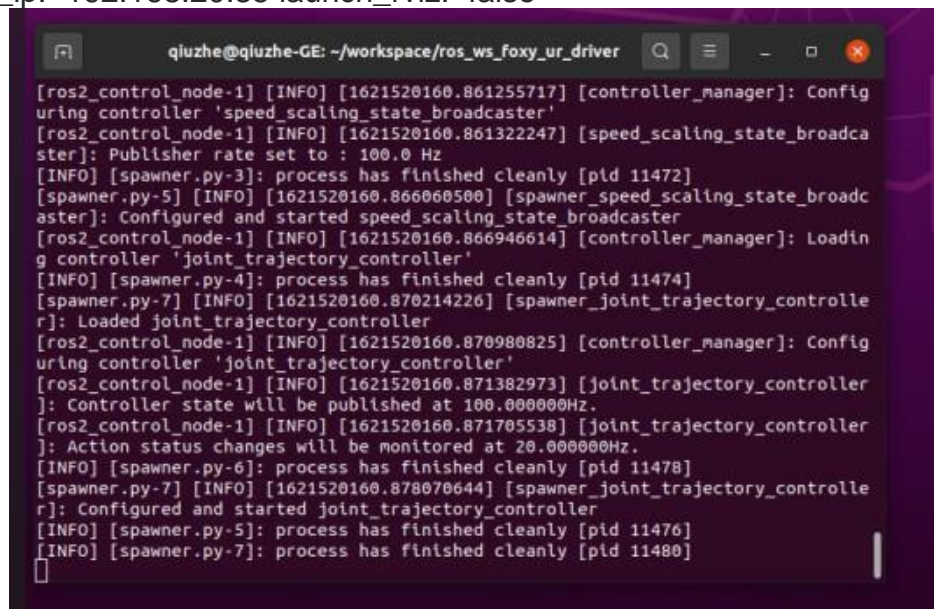
```
$ export COLCON_WS=~/.workspace/ros_ws_foxy_ur_driver
```

```
$ 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
```

A screenshot of a terminal window titled 'qiuzhe@qiuzhe-GE: ~/workspace/ros_ws_foxy_ur_driver'. The terminal displays the output of the command 'ros2 launch ur_bringup ur_control.launch.py ur_type:=ur5e robot_ip:=192.168.20.35 launch_rviz:=false'. The output shows several log messages from different ROS nodes, including 'ros2_control_node-1', 'controller_manager', 'speed_scaling_state_broadcaster', 'spawnner.py-3', 'spawnner_speed_scaling_state_broadcaster', 'spawnner.py-4', 'spawnner_joint_trajectory_controller', 'ros2_control_node-1', 'joint_trajectory_controller', and 'spawnner.py-5'. The messages indicate that the controllers are being configured and started successfully, with some mentioning 'process has finished cleanly' and others 'Configured and started'. The terminal window has a dark background and standard window controls at the top.

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:

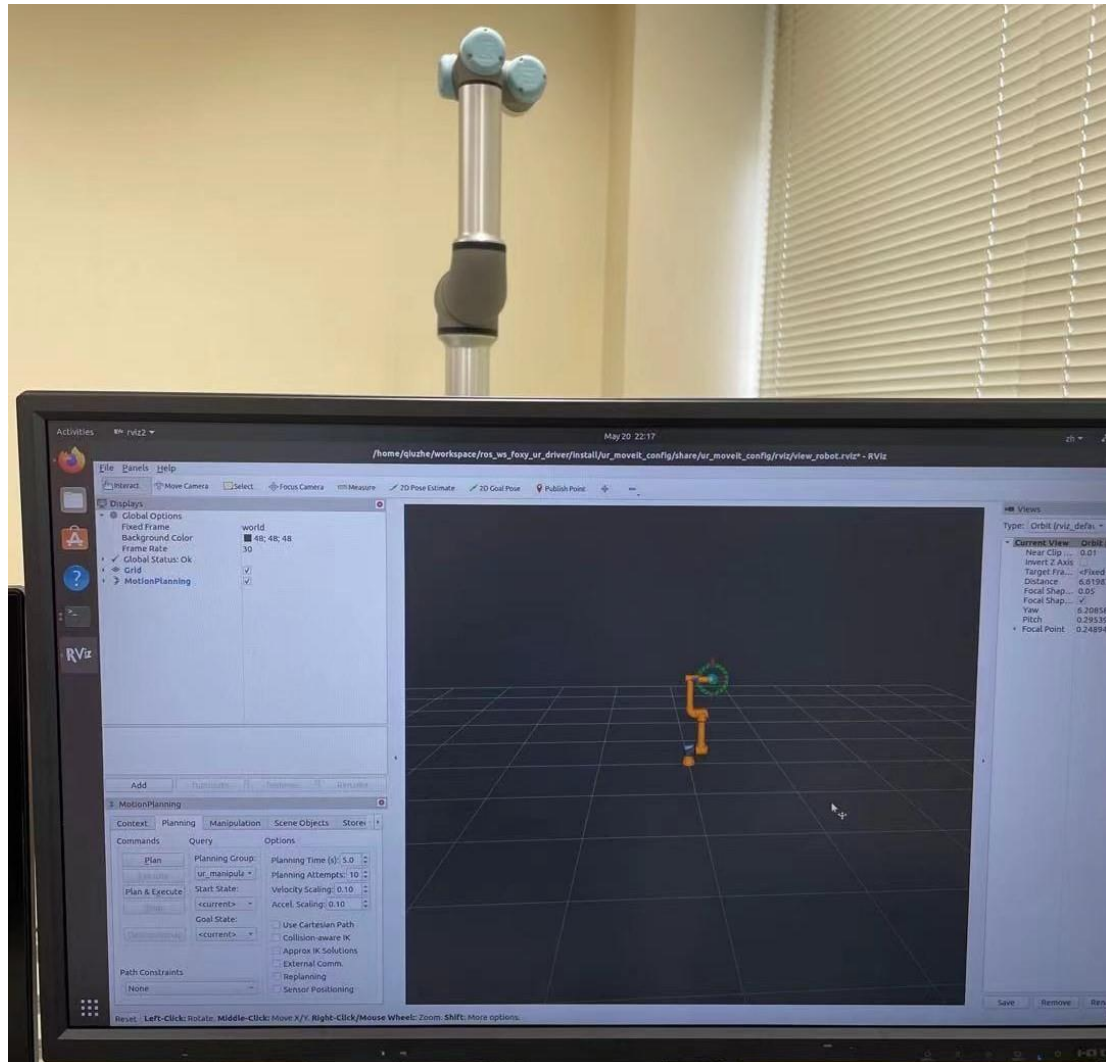
```
$ export COLCON_WS=~/.workspace/ros_ws_foxy_ur_driver
```

```
$ 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.

3.5 Modified ROS 2 package of scaled-/joint trajectory controller

A ur_5e robot is controlled via modified joint_trajectory_controller by using a PC (ROS 2 Foxy with Ubuntu 20.04). Similarly, the same setting can be applied to the scaled_joint_trajectory_controller for controlling the ur_5e robot.

The following steps are recommended: Step 1:

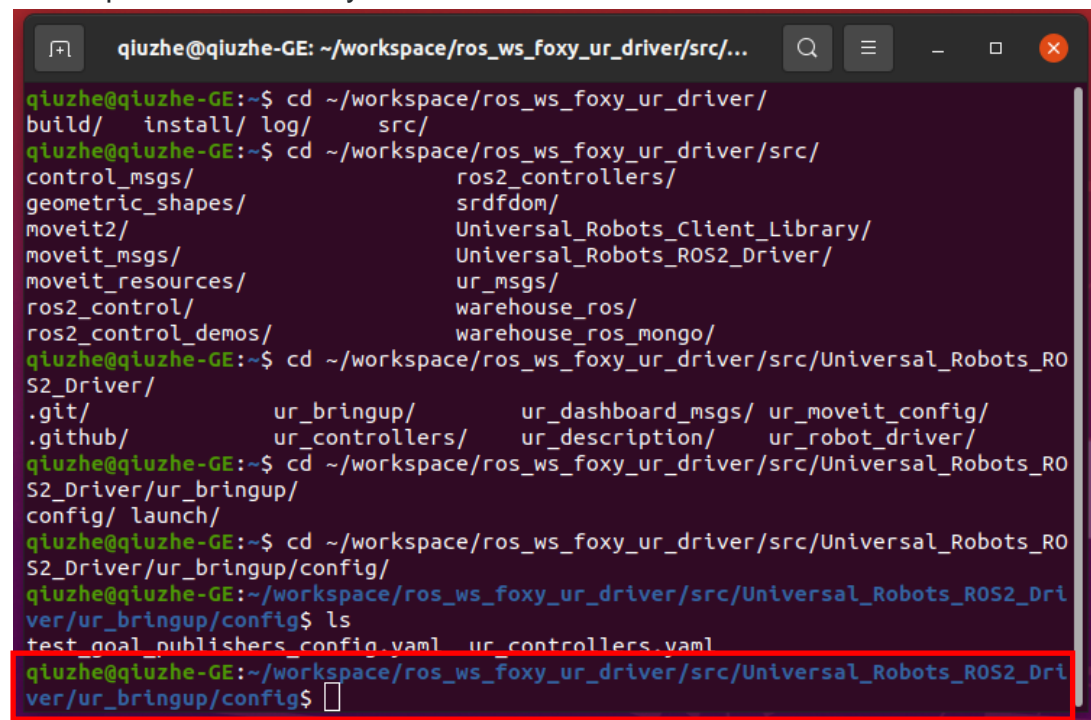
Set desired position of each joint

Open Terminal 1:

Find the controller config file.

\$ cd ~

/workspace/ros_ws_foxy_ur_driver/src/Universal_Robots_ROS2_Driver/

A terminal window screenshot showing a series of commands to navigate through a ROS2 workspace. The user starts at the home directory, moves to the workspace, then to the src directory, and finally to the Universal_Robots_ROS2_Driver directory. The terminal output shows the directory structure being traversed, including subdirectories like control_msgs, geometric_shapes, moveit2, moveit_msgs, moveit_resources, ros2_control, and ros2_control_demos. The final command is 'ls' in the directory ~/workspace/ros_ws_foxy_ur_driver/src/Universal_Robots_ROS2_Driver/ur_bringup/config/, which lists the files test_goal_publishers_config.yaml and ur_controllers.yaml. The last line of the terminal shows the user entering the vim editor to edit test_goal_publishers_config.yaml.

```
qiuzhe@qiuzhe-GE: ~$ cd ~/workspace/ros_ws_foxy_ur_driver/
build/  install/ log/      src/
qiuzhe@qiuzhe-GE:~$ cd ~/workspace/ros_ws_foxy_ur_driver/src/
control_msgs/      ros2_controllers/
geometric_shapes/  srdfdom/
moveit2/           Universal_Robots_Client_Library/
moveit_msgs/       Universal_Robots_ROS2_Driver/
moveit_resources/  ur_msgs/
ros2_control/      warehouse_ros/
ros2_control_demos/ warehouse_ros_mongo/
qiuzhe@qiuzhe-GE:~$ cd ~/workspace/ros_ws_foxy_ur_driver/src/Universal_Robots_ROS2_Driver/
.git/      ur_bringup/      ur_dashboard_msgs/ ur_moveit_config/
.github/   ur_controllers/  ur_description/   ur_robot_driver/
qiuzhe@qiuzhe-GE:~$ cd ~/workspace/ros_ws_foxy_ur_driver/src/Universal_Robots_ROS2_Driver/ur_bringup/
config/ launch/
qiuzhe@qiuzhe-GE:~$ cd ~/workspace/ros_ws_foxy_ur_driver/src/Universal_Robots_ROS2_Driver/ur_bringup/config/
qiuzhe@qiuzhe-GE:~/workspace/ros_ws_foxy_ur_driver/src/Universal_Robots_ROS2_Driver/ur_bringup/config$ ls
test_goal_publishers_config.yaml  ur_controllers.yaml
qiuzhe@qiuzhe-GE:~/workspace/ros_ws_foxy_ur_driver/src/Universal_Robots_ROS2_Driver/ur_bringup/config$ vim test_goal_publishers_config.yaml
```

ur_bringup/config

Modify the controller config file: change desired position of each joint.

\$ vim test_goal_publishers_config.yaml

Six values of “pos” mean the desired position associated with six joints. In addition, “pos1-4” mean four desired joint trajectories for the ur_5e robot.


```
qiuzhe@qiuzhe-GE: ~/workspace/ros_ws_foxy_ur_driver/src/...  
- wrist_2_joint  
- wrist_3_joint  
  
publisher_joint_trajectory_controller:  
  ros__parameters:  
    controller_name: "joint_trajectory_controller"  
    wait_sec_between_publish: 8  
    goal_names: ["pos1", "pos2", "pos3", "pos4"]  
    pos1: [0.785, -0.785, 0.785, 0.785, 0.785, 0.785]  
    pos2: [0.0, -1.57, 0.0, 0.0, 0.0, 0.0]  
    pos3: [-0.785, -0.785, 0.785, -0.785, -0.785, -0.785]  
    pos4: [0.0, -1.57, 0.0, 0.0, 0.0, 0.0]  
    joints:  
      - shoulder_pan_joint  
      - shoulder_lift_joint  
      - elbow_joint  
      - wrist_1_joint  
      - wrist_2_joint  
      - wrist_3_joint
```

Step2:

The package should be recompiled after modification In Terminal 1:

\$ cd ~/workspace/ros_ws_foxy_ur_driver

Only compile the modified package to save time

\$ colcon build --packages-select ur_bringup

```
qiuzhe@qiuzhe-GE: ~/workspace/ros_ws_foxy_ur_driver  
qiuzhe@qiuzhe-GE:~$ cd ~/workspace/ros_ws_foxy_ur_driver/  
qiuzhe@qiuzhe-GE:~/workspace/ros_ws_foxy_ur_driver$ colcon build --packages-select ur_bringup  
Starting >>> ur_bringup  
Finished <<< ur_bringup [1.60s]  
  
Summary: 1 package finished [3.19s]  
qiuzhe@qiuzhe-GE:~/workspace/ros_ws_foxy_ur_driver$
```

Then, refer to Step1 to Step6 of Sec 3.2 to control the ur_5e robot. The performances of the joint trajectory controller are shown as follows.

