ROS2 Guide for Universal Robots

Davis Onyeoguzoro

Introduction

ROS2 - Robotics Operating System 2 is a set of open source robotic middleware and tools. It is the successor of ROS1. A key difference between ROS1 and ROS2 is that ROS1 uses the Master-Slave Architecture and the XML-RPC middleware, while ROS2 uses Data Distribution Service (DDS), which is designed to provide higher efficiency and reliability, low latency, and scalability, as well as configurable quality of service (QoS).

ROS2 Distributions

A ROS distribution is a versioned set of ROS packages. The purpose of having several distributions is to let developers work against a relatively stable codebase until they are ready to roll everything forward. Below are some of ROS2 distributions:

1. Jazzy Jalisco
2. Iron Irwini
3. Humble Hawksbill
4. Foxy Fitzroy
5. Dashing Diademata

There are many ROS2 distributions, and you can find the complete list here including their respective release dates. However, we will be making use of humble as our ROS2 distribution.

Installation

System Configuration

ROS2 supported Operating Systems:

1. Windows - ROS2 can be installed on Windows but it is crucial to have VisualStudio2019 and CMake 3.28 or above. The Windows version is important, so check the version, as some distributions would not work on some certain versions, there is usually support for Windows 10 and 11. Kindly refer to this link and follow the guide on Installing ROS2 on your Windows Operating System. Also, take note of the ROS2 distribution you are installing.
2. MacOS - ROS2 can be installed on Macbooks running MacOS, the current supported versions as at the time of writing this are OS X El Capitan and maOS Sierra (10.11.x and 10.12.x). In other to install ROS2, there are a few dependencies,
    - Homebrew: Most Macbooks come with brew installed, to check if brew is installed, use the command below in the terminal to check the version installed;

```
brew -v
```

if it returns "zsh: command not found: brew
Then follow the instructions in this link to install brew. Use brew to install the remaining ROS2 dependencies

- Remaining dependencies;

```
brew install asio tinyxml2
brew install tinyxml eigen pcre poco
brew install openssl
brew install qt freetype assimp
brew install log4cxx
brew install sip pyqt5
```

3. Linux - Ubuntu is the most common and well-supported platform for ROS2. Ubuntu 22.04 (Jammy Jellyfish) is recommended for the Humble distribution.

In this guide, we would be making use of Ubuntu 22.04 (Jammy Jellyfish).

Prerequisites: Ensure your system us up-to-date by running the following commands:

```
sudo apt update && sudo apt upgrade -y
sudo apt install curl gnupg lsb-release -y
```

Add the ROS2 Repository:

```
sudo apt update && sudo apt upgrade -y
sudo apt install software-properties-common
sudo add-apt-repository universe
sudo apt update
```

Install ROS2 Humble:

```
sudo apt update
sudo apt install ros-humble-desktop
```

Setup Environment:

```
echo "source /opt/ros/humble/setup.bash" >> ~/.bashrc
source ~/.bashrc
```

Test Installation:

```
ros2 --version
```

Download and build ROS2 packages

Step 1:
source global ROS2, it is very important to source ROS2

```
sudo nano ~/.bashrc
```

Add this line to the last line of your opened bashrc file

```
source /opt/ros/humble/setup.bash
```

Now save the file, after adding the line.
Source the bash file, using the command below;

```
source ~/.bashrc
```

Step 2: Workspace
Create a ROS2 workspace: A workspace is simply a directory that contains ROS2 packages. It is a directory that houses all the different software packages, data files, and configuration scripts related to your specific robotics project.

```
mkdir -p MY_WS/src
```

Step 3: Colcon
colcon build is used to build a ROS2 workspace, it compiles your code and generates executables, libraries and other artifacts.

```
cd MY_WS
git clone https://github.com/UniversalRobots/Universal_Robots_ROS2_Driver
rosdep install --ignore-src --from-paths src -y -r
colcon build --cmake-args -DCMAKE_BUILD_TYPE=Release
source install/setup.bash
```

Setting Up Universal Robots (UR5e) with ROS2

1.1 Activate Robot

Check the Robot Serial Number at the back of the Universal Robot. Power on the UR (Universal Robot) using the TP (Teaching Pendant). Go to Settings by clicking the hamburger icon on the top right. Click on Licences and enter the UR default password which is easybot. Now visit the official site. Fill in the necessary details, after clicking on Register Robot, you would be prompted to download a license file, download it, then upload/copy this downloaded file (license.p7b) to a flash or hard drive. Put the drive into the Control Box and upload the file in the License. You can also change Admin password, it is adviceable to do so.

1.2 Setup the Universal Robot

To control the robot from your PC, you need to have external control installed on the robot. Download the latest release of the externalcontrol.urcap from this link. At the time of this writing, the latest version is 1.0.5.

1. For installing this URCap, a minimal PolyScope version 5.1 (for e-Series) is necessary.
2. When I refer to robot, I mean the universal robot (any of the UR series - ur3, ur5e, others).
3. TP - Teach Pendant

1.3 URCap Installation

To install URCap on the UR robot. Follow the following steps;

1. Copy the URCap (external control) in any external storage device.
2. On the TP of the UR robot, click the hamburger menu in the top-right corner and select Setttings to enter the robot's setup. Select System and then URCaps to enter the URCaps installation screen.
3. On the screen, click the plus sign at the bottom to open the file selector. This will open all URCap files stored inside the robots program folder. Select and open the externalcontrol.urcap file. This would install the External Control in your UR Robot. Restart the UR Robot.

   Go to the Teach Pendant → System Setup → URCaps Setup.
   Confirm that the URCap "ExternalControl" is installed.
4. After the UR robot is restarted, you will find the External Control in URCaps in the Installation tab in your home menu (TP menu).
5. Network Configuration - Ensure the robot is connected to the same network as your PC. You need to setup the IP Address of your PC, that is going to be running the ROS2 driver. The UR robot and your PC "must" be in the same network, else you can not control the robot, checking

the network configuration of the PC running ROS2 and the robot is the first step to debugging or finding causes of errors. It is also advised to use a direct connection between the PC and the UR robot to minimize network disturbances.

6. Your UR robot is now ready to be used together with the ROS2 driver in your PC.

ROS2 Driver

There will be two (2) ways to use ROS2 to control the Universal Robots;

1. Official ROS2 Driver by Universal Robot
2. Custom ROS2 Driver currently being developed as at the time of writing (14th February, 2025)

Controlling the UR Robot with ROS2

In this section of this tutorial, we will be controlling the Universal Robot using the ROS2 driver developed by Universal Robot.

Installing ROS2

ROS2 should be installed on your PC, if not, go to the Installation section in this same guide, and follow the steps I outlined there.

Prepare the robot

Power ON the robot

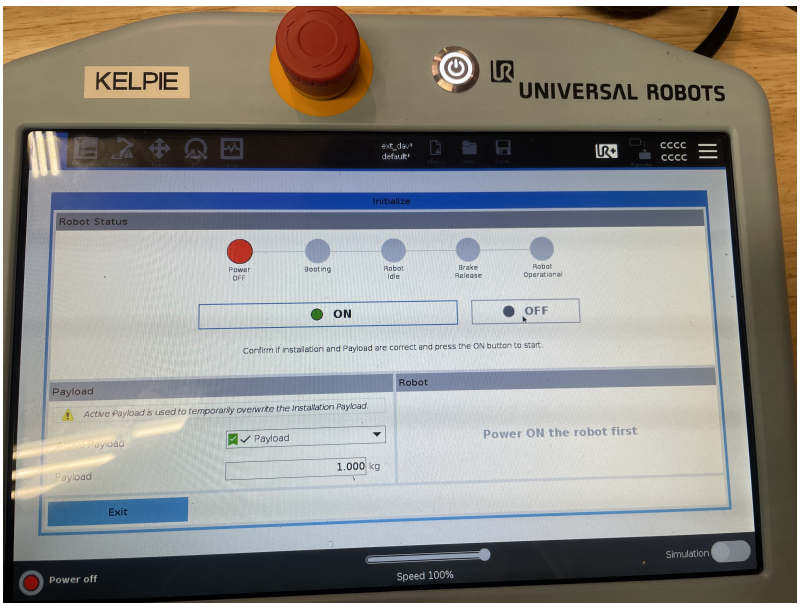

Figure 1: Power ON the UR Robot

Start the UR Robot

Depending on the series of your UR Robot, you might be prompted to set a PAYLOAD before starting your UR Robot, either way, it is important to set a payload of the UR Robot (e.g 1.0kg). At this time your real-time UR Robot position will be displayed at the bottom-right corner of the Teach Pendant (as shown in the image below).

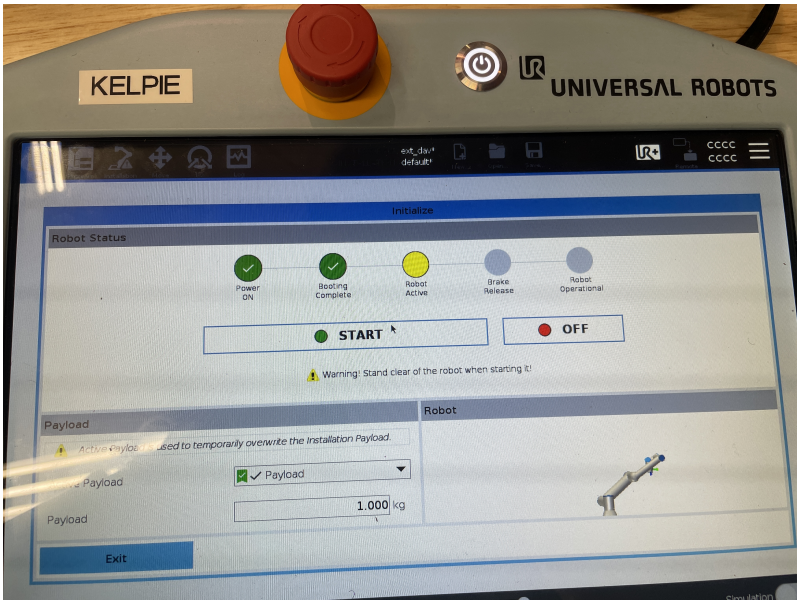

Figure 2: Start the UR Robot

Robot is Active

The UR Robot is Active and Ready. All the boxes should be checked as shown in the image.

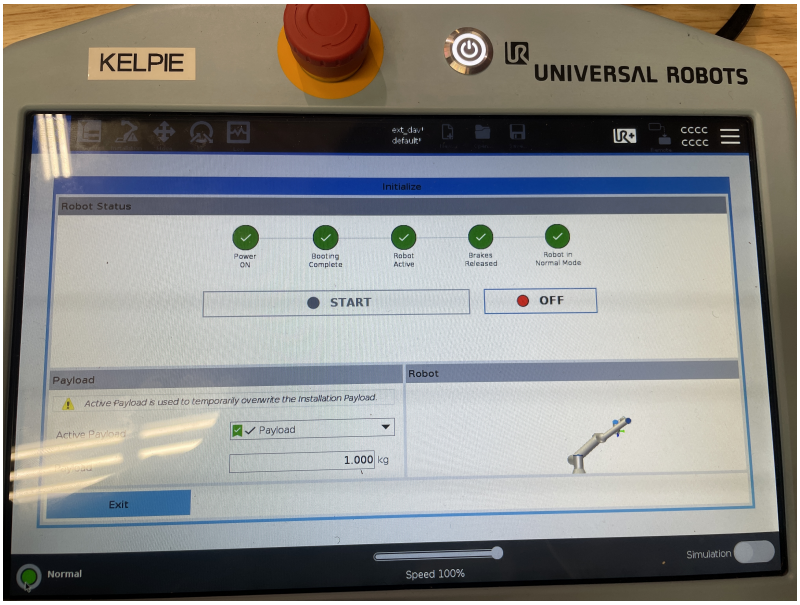

Figure 3: Robot is Ready

Network Configuration

Ensure that your PC and the UR Robot is in the same network, preferably via a network cable, but WiFi works as well.

Set the IP Address of the UR robot

Go the Settings on the UR TP and configure the IP Address to Static Address, for instance: 129.101.98.211

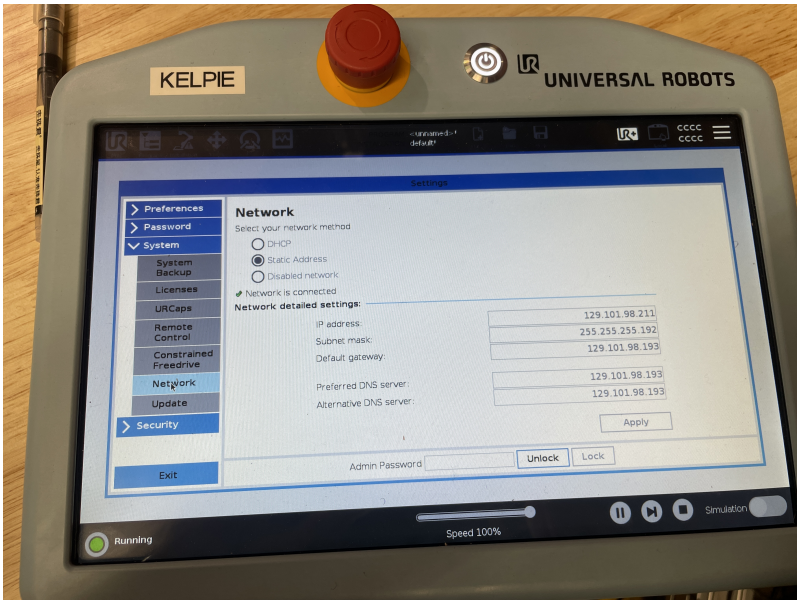

Figure 4: UR Robot IP

Set the IP Address of your PC

Check the IP Address of your PC, and note it down, you can check your IP Address using;
Windows:

```
ipconfig
```

Mac and Linux:

```
ifconfig
```

Set IP Address on External Control

Go to the Installation Tab on your UR Robot TP, and set the Host IP and Host name, both should be the IP Address of your PC

Test the connection

Ping the UR Robot to confirm that your PC and the UR Robot are on the same network.

```
ping <UR-Robot-IP>
```

Example

```
ping 129.101.98.5
```

Control Mode

There are two modes in the UR Robot.

1. Local Mode - Controlling the UR Robot directly through the Teach Pendant
2. Remote Control - Controlling the Robot from external sources (e.g PC)

Set the UR Robot control mode to Local Mode on the Teach Pendant, (as shown in the top right corner of the image).

Figure 5: Local Mode

The steps below will show how to operate the UR Robot using ROS2

Open Terminal

Enter the ROS2 workspace that we created in the Installation section of this tutorial.

```
cd MY_WS
```

Source the Workspace

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

Launch ROS2

Launch the robot control node:

```
ros2 launch ur_robot_driver ur_control.launch.py ur_type:=ur5e
    ur_ip:=<ROBOT_IP>
```

Sample code:

```
ros2 launch ur_bringup ur_control.launch.py ur_type:=ur5e
    robot_ip:=129.101.98.5 launch_rviz=true
```

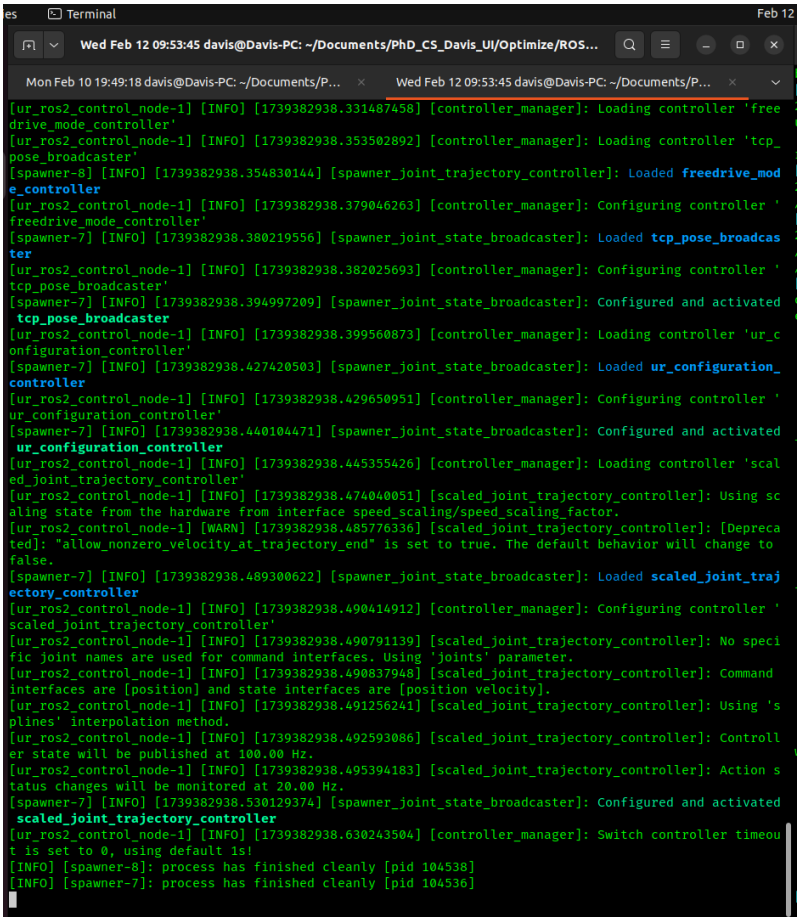

Figure 6: Loaded all files

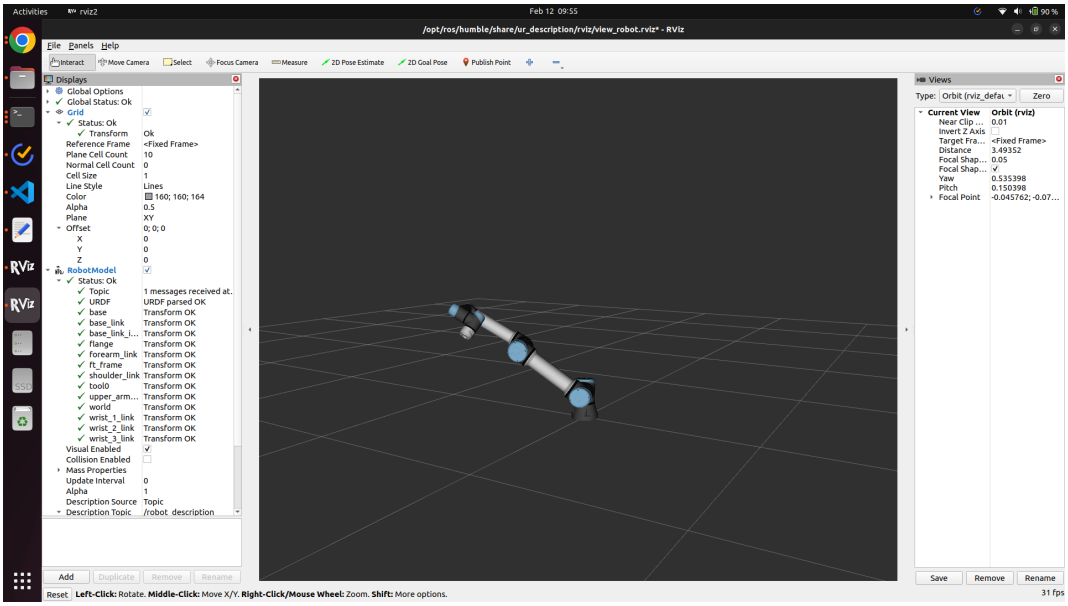

Figure 7: Simulation of Real UR Robot

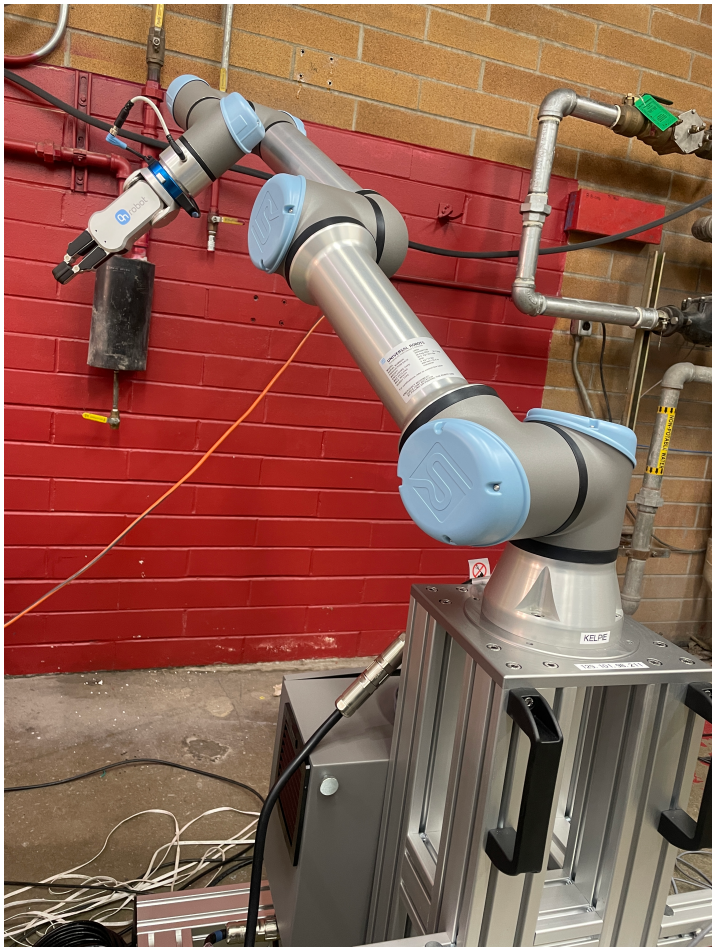

Figure 8: Real-time UR Robot

Load Robot Program

Start the program that contains your laptop IP Address programmed on the UR Robot

Move the Robot from the terminal

```
ros2 topic pub /scaled_joint_trajectory_controller/joint_trajectory
    trajectory_msgs/JointTrajectory
"joint_names:
  - 'shoulder_pan_joint'
  - 'shoulder_lift_joint'
  - 'elbow_joint'
  - 'wrist_1_joint'
  - 'wrist_2_joint'
  - 'wrist_3_joint'
points:
  - positions: [0.0, -1.57, 1.57, 0.0, 0.0, 0.0]
    time_from_start: {sec: 3, nanosec: 0}"
```

Observe the UR Robot move

Now wait and observe the robot move to the new position.

It is important to note that the robot uses radians and not degrees, if your measurement is in the degrees, you have to convert to radians, and vice versa.

Troubleshooting

Common Issues

- Driver Connection Fails: Ensure the IP configuration is correct and that the robot is in "Remote Control" mode.
- Unresponsive ROS2 Nodes: Verify that all ROS2 nodes are running correctly and that the ROS2 workspace is sourced.

Debugging Tools

- Use ros2 topic list to ensure all required topics are available.
- Use rviz2 to visualize the robot's state and environment.

How to Build ROS2 Drivers

Make Directory or Workspace for the driver:

```
mkdir -p ros2_ws/src
cd ros2_ws
```

Colcon build is a command line tool that builds a set of packages in the ROS. For compiling and configuring.

```
colcon build
```

Source the workspace

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

Packages: To create a ROS Node, you need a package, Packages allow codes to be separated, they are independent unit.

Create Package

```
cd ros2_ws/src
ros2 pkg create msg_publishers --build-type ament-python --dependencies rclpy
```

```
cd ros2_ws/src
colcon build
```

Nodes: A Node is a subpath of the ROS2 application, and should have a single purpose. Nodes communicate with each other, and are created inside packages. Each node in ROS should be responsible for a single, modular purpose.

Create Package

```
cd ros2_ws/src
ros2 pkg create msg_publishers --build-type ament-python --dependencies rclpy
```

```
cd ros2_ws/src
colcon build
```