Introduction to ROS - The Robot Operating System – PART 3

Version	Date	Author	Comments
1.0	Jan 2018	Arthur Richards	New Part 3 on UR arm
1.1	Feb 2018	Arthur Richards	Added extra arm training information
1.2	Mar 2019	Arthur Richards	Extra information on dependency installations Altered UR set-up for direct connection to arm.

This handout shows how to use ROS to drive a Universal Robotics UR10 robot arm, like those mounted on the Neobotix MPO-700 platforms at BRL. Procedures for UR5 or UR3 arms are almost identical.

You can get code for this tutorial from https://github.com/arthurrichards77/ur_tutorial

1 Software Installation

The ROS drivers and simulators for the UR family of arms are provided in the universal_robot package. We're using ROS Melodic in the BRL PC lab in 2019 and this particular package hasn't been upgraded since Kinetic, so you'll need to install it from source by following these simple steps:

1. In a terminal, navigate to the 'src' subdirectory of your ROS catkin workspace, which you should have set up in our first ever training session.
2. Download the source code from Github by typing:

```
git clone https://github.com/ros-industrial/universal_robot
```

3. Navigate back to the root directory of your workspace, above 'src', and compile it using the command below. You may get some errors about missing 'moveit' content but that's OK. If you get any other errors, please seek help.

```
catkin_make
```

4. With `rospack list`, check you have the "ur_gazebo" package installed.

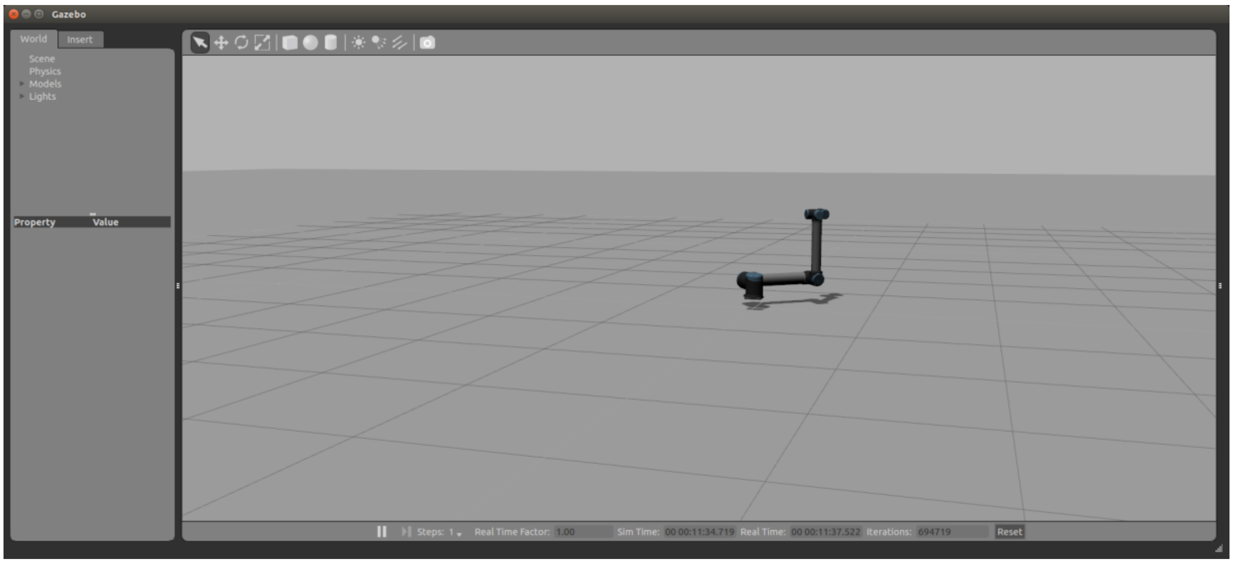
2 Simulated Arm

The “ur_gazebo” package provides a standard simulator for UR arms with the same behaviour and interface as the real arm. Run it using:

```
roslaunch ur_gazebo ur10_joint_limited.launch
```

You should see the Gazebo simulator with a world populated only by an arm.

> *Note: you’ll also get some red error messages about missing proportional gains in the controller descriptions. You are safe to ignore these. If you get any other errors, e.g. ‘*`unable to start arm_controller`*’ check that the ros_controllers packages as installed and that you have ‘*`joint_trajectory_controller`*’ showing in your* `rospack list`*. If not, summon help.*



Now use `rqt_graph`, `rostopic` and `rosparam` to have a poke around the interface. The `joint_states` topic and `robot_description` parameter are of interest.

Create the following python script in the “scripts” subdirectory of your “ros_course” folder (or go and find it in “ur_tutorial”). This publishes a trajectory command to the robot controller, by using the “goal” topic of its “follow joint trajectory” action. (See ROS documentation for more detail on how to make the most of “actions”.)

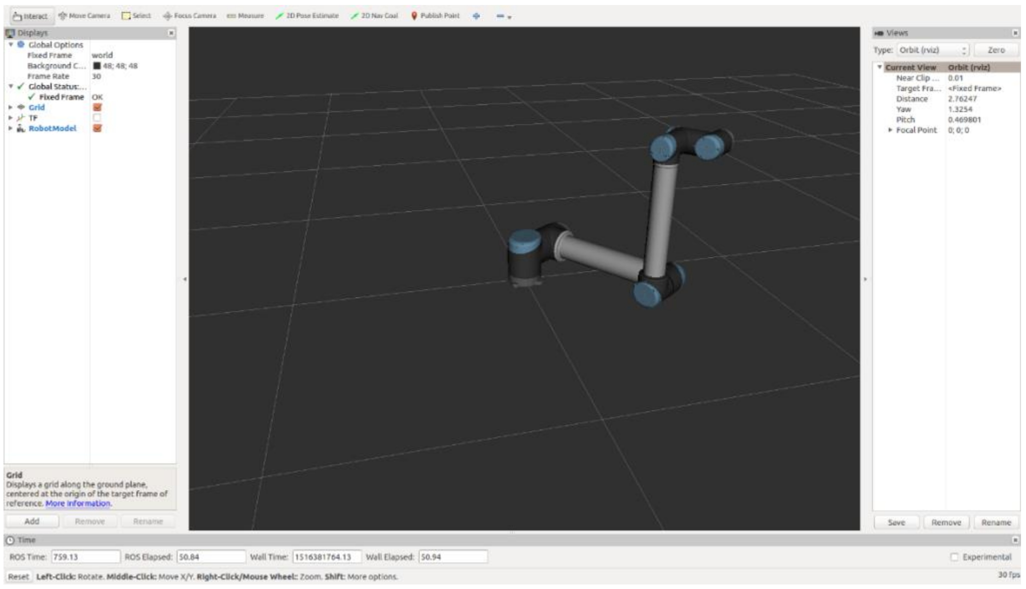
```
#!/usr/bin/env python
import roslib
roslib.load_manifest('ros_course')
import rospy
from trajectory_msgs.msg import JointTrajectory,JointTrajectoryPoint
from control_msgs.msg import FollowJointTrajectoryActionGoal

rospy.init_node("simple_move_goal_pub")
pub = rospy.Publisher("/arm_controller/follow_joint_trajectory/goal",
FollowJointTrajectoryActionGoal, queue_size=10)
rospy.sleep(0.5)

traj = JointTrajectory()
traj.joint_names = ['shoulder_pan_joint', 'shoulder_lift_joint',
'elbow_joint', 'wrist_1_joint', 'wrist_2_joint', 'wrist_3_joint']

now = rospy.get_rostime()
rospy.loginfo("Current time %i %i", now.secs, now.nsecs)
traj.header.stamp = now

p1 = JointTrajectoryPoint()
p1.positions = [1.5, -0.2, -1.57, 0, 0 ,0]
p1.velocities = [0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
p1.time_from_start = rospy.Duration(5.0)
traj.points.append(p1)

p2 = JointTrajectoryPoint()
p2.positions = [1.5, 0, -1.57, 0, 0 ,0]
p2.velocities = [0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
p2.time_from_start = rospy.Duration(10.0)
traj.points.append(p2)

p3 = JointTrajectoryPoint()
p3.positions = [2.2, 0, -1.57, 0, 0 ,0]
p3.velocities = [0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
p3.time_from_start = rospy.Duration(15.0)
traj.points.append(p3)

ag = FollowJointTrajectoryActionGoal()
ag.goal.trajectory = traj

pub.publish(ag)
```

Run it, and you should see your simulated arm moving. Experiment with different movements.

3 Visualizing with Rviz

With the robot simulator still running, start Rviz.

If it's not already running, add a "RobotModel" widget using the "Add" button at the bottom left. You should be able to see your UR10 robot in Rviz now.



Displays

Global Options	
Fixed Frame	world
Background C...	48; 48; 48
Frame Rate	30
✓ Global Status:...	
✓ Fixed Frame	OK
Grid	✓
TF	☐
RobotModel	✓

Grid
Displays a grid along the ground plane, centered at the origin of the target frame of reference. More information.

Views

Type: Orbit (rviz)

Current View	Orbit (rviz)
Near Clip ...	0.01
Target Fra...	<Fixed Frame>
Distance	2.76247
Yaw	1.3254
Pitch	0.469801
Focal Point	0; 0; 0

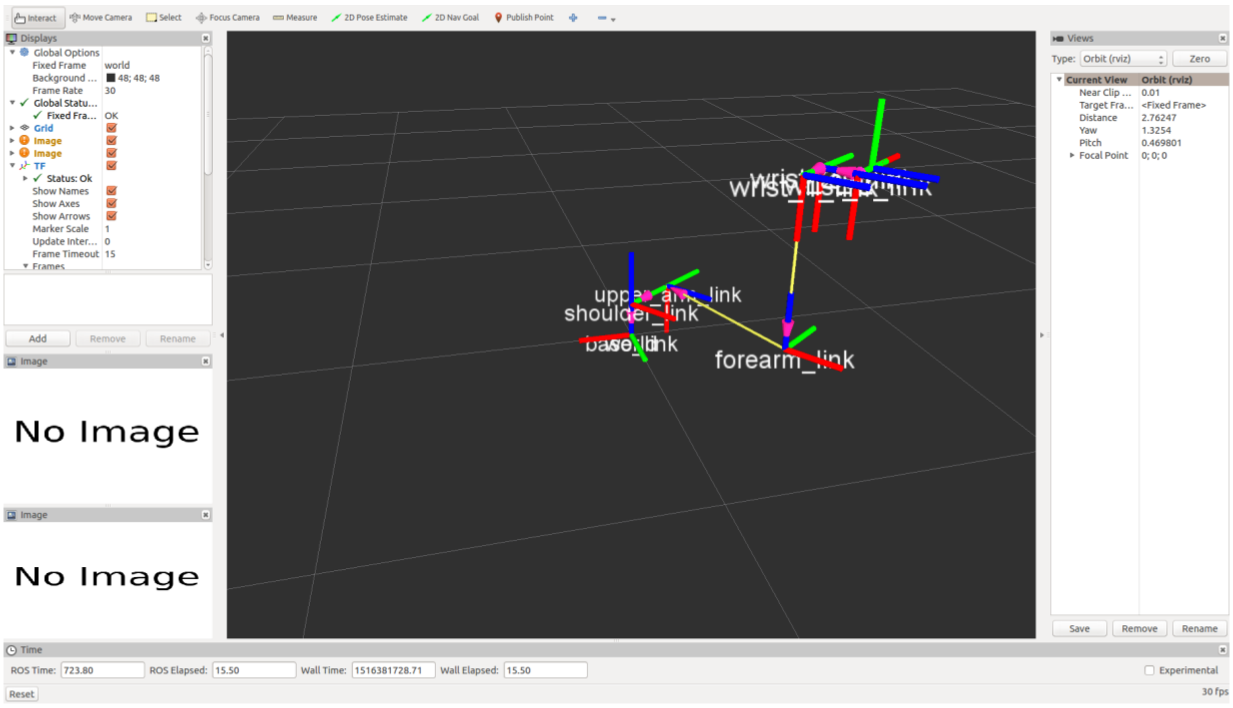
Time

ROS Time: 759.13 ROS Elapsed: 50.84 Wall Time: 1516381764.13 Wall Elapsed: 50.94 ☐ Experimental

Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click/Mouse Wheel:: Zoom. Shift: More options. 30 fps

Note: Rviz and Gazebo now look rather similar. Rviz is just drawing your robot using the "joint_states" and "robot_model" information from Gazebo. Gazebo is simulating its behaviour.

Then click the "add" button on the lower left and add a new TF viewer. It's easier to see what's going on if you disable the robot model. This is the transform tree for the robot reference frames, derived from the URDF in the "robot_model" parameter. The "robot_state_publisher" node uses the URDF and the "joint_states" information to "tf" transforms. You could use this information to find the pose of a target relative to the end effector.



Displays

Global Options	
Fixed Frame	world
Background ...	48; 48; 48
Frame Rate	30
✓ Global Statu...	
✓ Fixed Fra...	OK
Grid	✓
Image	✓
Image	✓
TF	✓
✓ Status: Ok	
Show Names	✓
Show Axes	✓
Show Arrows	✓
Marker Scale	1
Update Inter...	0
Frame Timeout	15
Frames	

Views

Type: Orbit (rviz)

Current View	Orbit (rviz)
Near Clip ...	0.01
Target Fra...	<Fixed Frame>
Distance	2.76247
Yaw	1.3254
Pitch	0.469801
Focal Point	0; 0; 0

Time

ROS Time: 723.80 ROS Elapsed: 15.50 Wall Time: 1516381728.71 Wall Elapsed: 15.50 ☐ Experimental

30 fps

4 Working with the UR10 arm

Caution: whenever moving the arm, have an operator ready on the red stop button on the console. It is possible to drive the arm into yourself or itself.

This section describes how to operate the UR10 arm mounted on the MPO700 base from Neobotix. Operating the movable base is not covered.

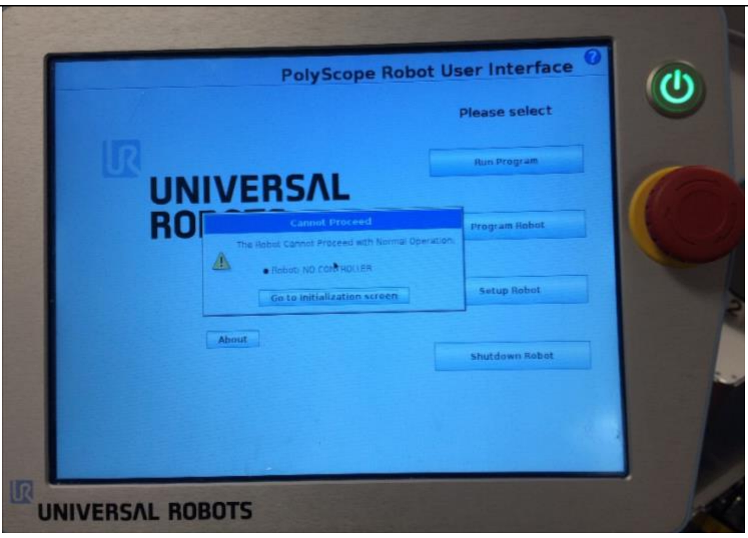
4.1 Turn on the arm

Press the power button above the red button on the pendant. The pendant screen will come on and run a boot process taking about a minute. You should get the message shown. Press "Go to initialization"

If this doesn't work, please check that your robot is plugged in and switched on at the mains.

☹

Robot 2 has a dodgy touch screen – please use the mouse connected to the pendant.



Cannot Proceed

The Robot Cannot Proceed with Normal Operation:

- Robot: NO CONTROLLER

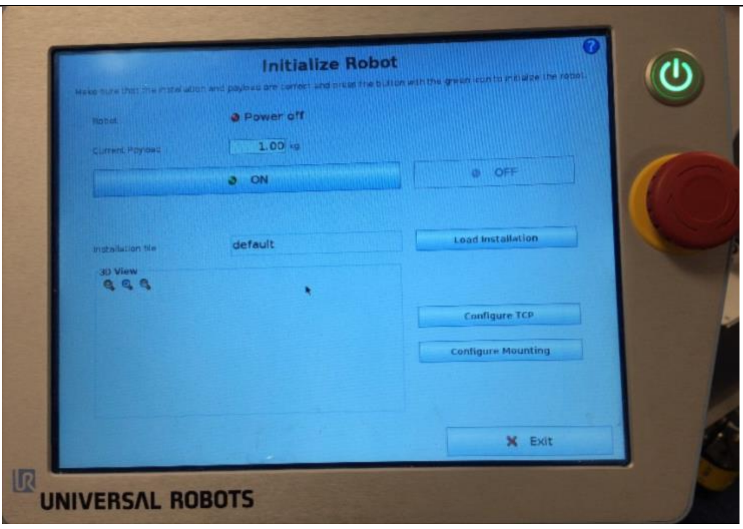
Go to initialization screen

4.2 Initialize the robot

Enter the appropriate payload value (it's not too sensitive – you can leave it at 1kg for a bare arm) and press "On". That button will then change to "Start".

Caution: robot lurches at startup. Warn others around you and ensure no-one is close to the arm.

Press the "Start" button and then "Exit".



Make sure that the installation and payloads are correct and press the button with the green icon to initialize the robot.

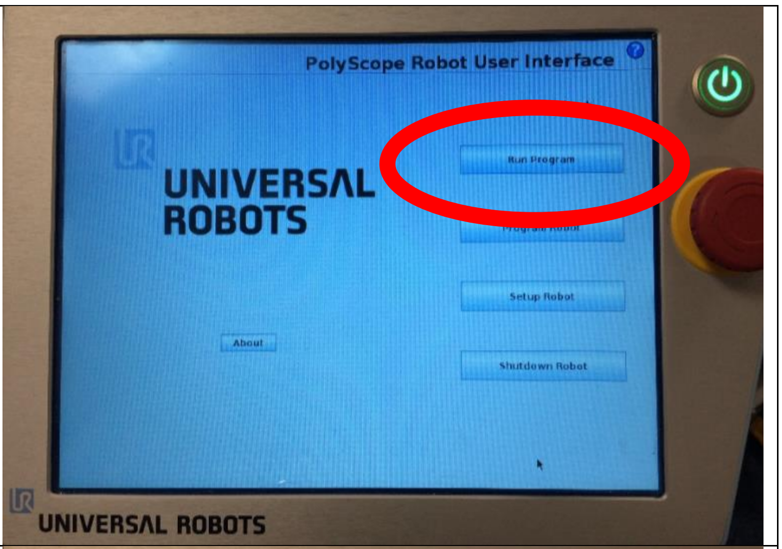
Robot: Power off

Current Payload: 1.00 kg

Installation file: default

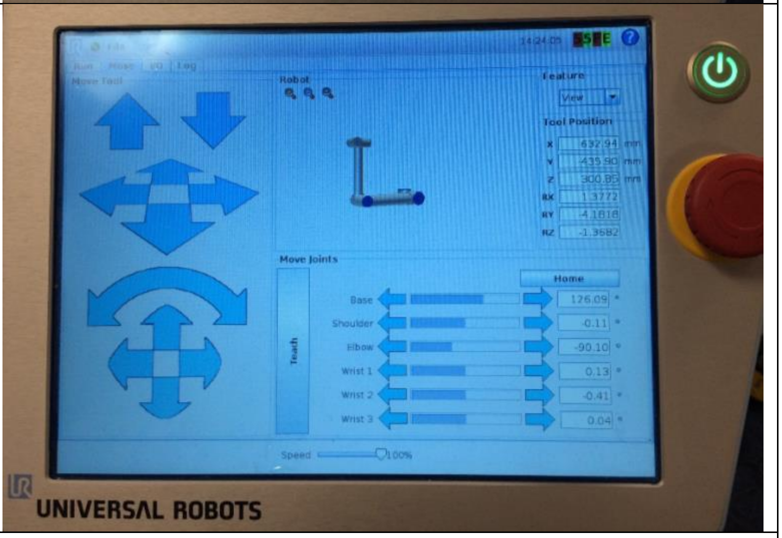
4.3 Move the arm manually

On the pendant main menu, press “Run Program”.



Select the “Move” tab and use the arrows to drive the arm. The left-hand arrows perform movements in the reference frame of the end effector. The right-hand arrows move joints individually.

Caution: end-effector moves can be hard to predict. Watch for singularities.



4.4 Turn off the robot

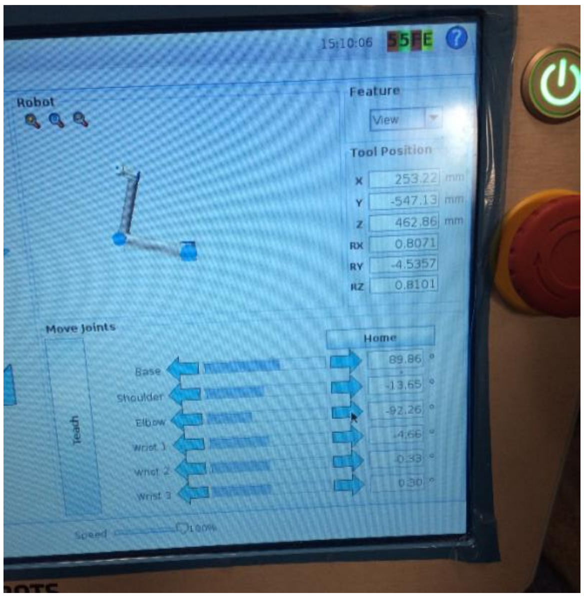
When you’re finished, press and hold the power button on the pendant to turn off. For the avoidance of errors, also press the emergency stop button and turn off at the mains.

5 Drive the arm from ROS

The real arm uses the same interface as the simulator but running on a different computer, onboard the MPO-700. We will use ROS communications to access that computer and then your code from earlier to command a movement.

5.1 Position the Robot

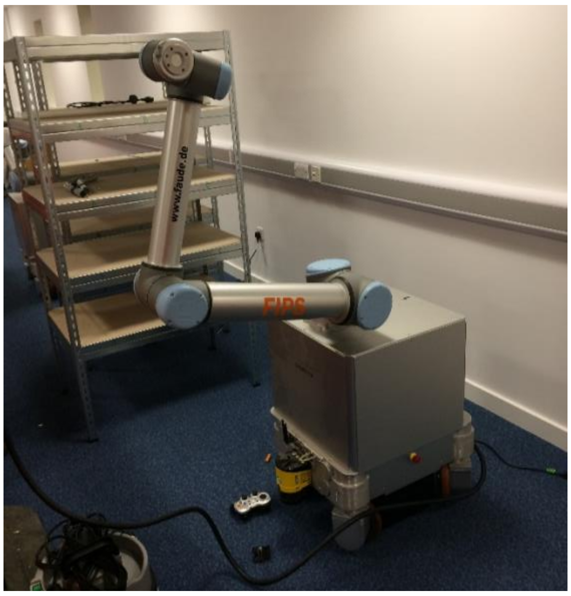
Using the manual controls on the UR10 pendant, drive the arm to approximately the position shown. Pay close attention to the joint values and exact position of stickers and connectors. There are other positions that look a bit like this but with totally different joint angles.



Tool Position

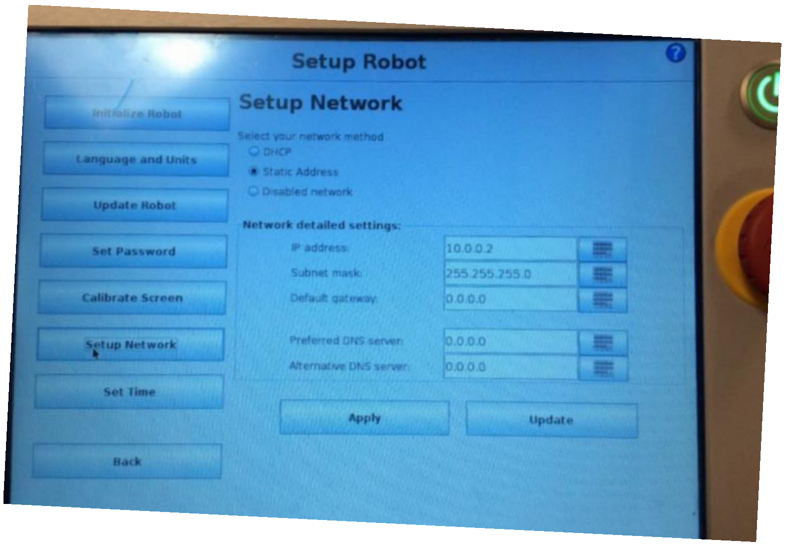
X	253.22	mm
Y	-547.13	mm
Z	462.86	mm
RX	0.8071	
RY	-4.5357	
RZ	0.8101	

Base	89.86 °
Shoulder	-13.65 °
Elbow	-92.26 °
Wrist 1	-4.66 °
Wrist 2	0.33 °
Wrist 3	0.30 °



5.2 Set up the communications

1. Join your computer to the router associated with the robot you're going to drive. It should give you an IP address in the 192.168.1.xxx range. Check using ifconfig.
   2. Find the IP address of the robot from the pendant using the "Setup Robot" and "Setup Network" screens (see below). For 2019, the address should be 192.168.1.RRR.



Setup Network

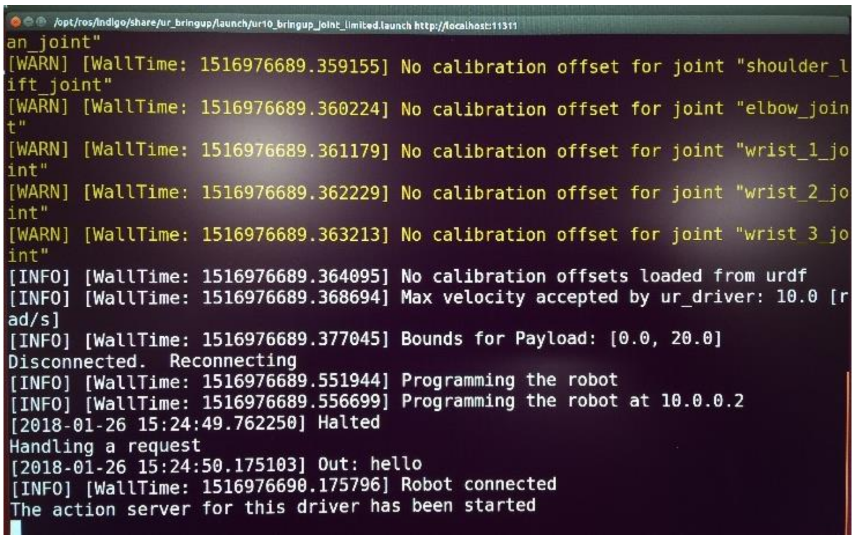
Select your network method
DHCP
[✓ Static Address]
Disabled network

Network detailed settings:

IP address:	10.0.0.2
Subnet mask:	255.255.255.0
Default gateway:	0.0.0.0
Preferred DNS server:	0.0.0.0
Alternative DNS server:	0.0.0.0

3. Check you can "`ping 192.168.0.RRR`" (the robot) from your computer
4. At the prompt, type the following to launch the interface:

```
roslaunch ur_bringup ur10_bringup_joint_limited.launch
robot_ip:=192.168.1.RRR
```



```
/opt/ros/indigo/share/ur_bringup/launch/ur10_bringup_joint_limited.launch http://localhost:11311
an_joint"
[WARN] [WallTime: 1516976689.359155] No calibration offset for joint "shoulder_l
ift_joint"
[WARN] [WallTime: 1516976689.360224] No calibration offset for joint "elbow_join
t"
[WARN] [WallTime: 1516976689.361179] No calibration offset for joint "wrist_1_jo
int"
[WARN] [WallTime: 1516976689.362229] No calibration offset for joint "wrist_2_jo
int"
[WARN] [WallTime: 1516976689.363213] No calibration offset for joint "wrist_3_jo
int"
[INFO] [WallTime: 1516976689.364095] No calibration offsets loaded from urdf
[INFO] [WallTime: 1516976689.368694] Max velocity accepted by ur_driver: 10.0 [r
ad/s]
[INFO] [WallTime: 1516976689.377045] Bounds for Payload: [0.0, 20.0]
Disconnected.  Reconnecting
[INFO] [WallTime: 1516976689.551944] Programming the robot
[INFO] [WallTime: 1516976689.556699] Programming the robot at 10.0.0.2
[2018-01-26 15:24:49.762250] Halted
Handling a request
[2018-01-26 15:24:50.175103] Out: hello
[INFO] [WallTime: 1516976690.175796] Robot connected
The action server for this driver has been started
```

5. In a new terminal, run "`rostopic list`" and "`rostopic echo /joint_states`" and check that you can see robot data and the action interface
6. Run RViz to get the robot visualization

5.3 Move the robot

Caution: whenever moving the arm, have an operator ready on the red stop button on the console. It is possible to drive the arm into yourself or itself.

Use the code from earlier to move the robot. You will need to use a remap to send the goal topic to the right place for the real robot. Avoid fast or large moves.

6 Checklist

1. Power on arm
2. Determine arm IP address
3. Warn bystanders and initialize robot
4. Ping robot
5. Launch arm driver
6. Check joint states
7. *Ready to run robot*
8. *In case of e-stop, go to 3.*
9. *In case of manual jog or comms loss, go to 4.*
10. E-Stop and power off arm

7 Troubleshooting

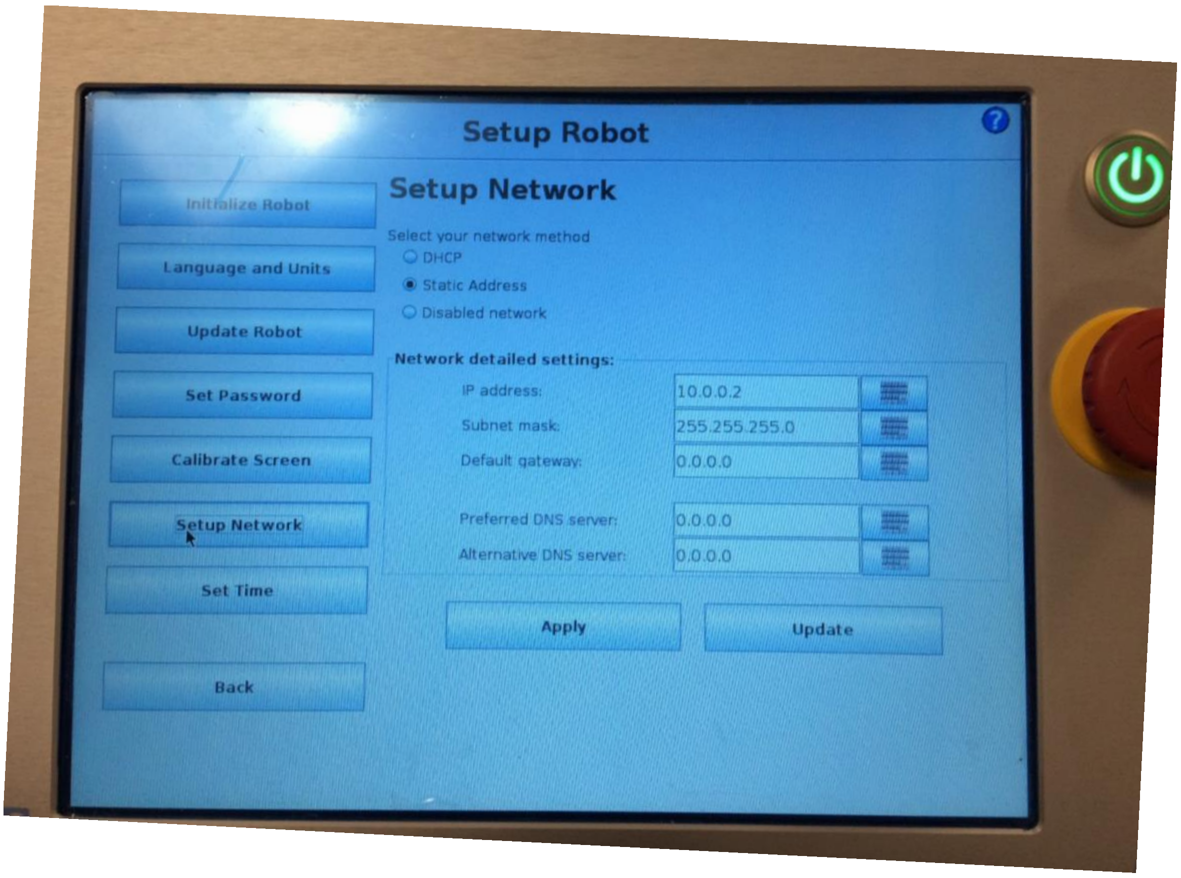
If you experience a loss of communication with the arm, test by doing “`rostopic echo /joint_states`”.

If you lose connection, try restarting the arm ROS driver: press Ctrl+C to shut down the roslaunch, and run the roslaunch again.

If you manually move the robot at any time, you will need to restart the arm ROS driver as above.

If you hit a singularity or use the emergency stop, you will need to repeat everything from “Initialize the Robot” onwards, including stopping and restarting the ROS interface.

In case the arm ROS driver fails to start, check the arm network settings.



Setup Network

Select your network method
DHCP
[✓ Static Address]
Disabled network

Network detailed settings:

IP address:	10.0.0.2
Subnet mask:	255.255.255.0
Default gateway:	0.0.0.0
Preferred DNS server:	0.0.0.0
Alternative DNS server:	0.0.0.0