# RBOT 250 Robot Manipulation, Planning and Control

Final Assignment: Setup Robot Arm in Gazebo

**By Salinee Kingbaisomboon**

**Sage ID: 20801897**

## Summary

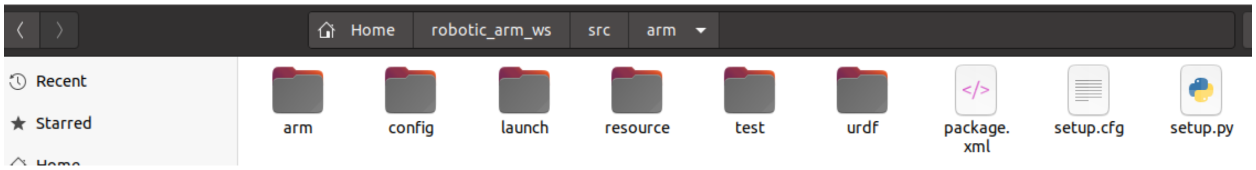
1. **Environment:**

   Below is the list of my setup used for this assignment:

   - VirtualBox
   - Ubuntu 20.04 focal
   - Ros 2 Foxy

2. **Important ROS2 Package:**

   All the code is under the folder called "robotic_arm_ws" which is the ROS workstation. The package to control is robot arm is called "arm" which you can see the structure of this package as follow:



   There are two important folders:

   1. Launch: This folder will store a launch file that I will place all command & ROS's nodes to open Gazebo, spawned a robot to it and publish the robot state publisher. The robot state publisher helps you to broadcast the state of your robot to the tf transform library.
   2. urdf: This folder contains an XML file format used in ROS to describe all elements of a robot. The important aspect of this URDF to work on Gazebo is that we need to add physical and collision properties to our URDF model. This link (http://gazebosim.org/tutorials/?tut=ros_urdf) provided me with a very good foundation of how to construct and add gazebo plugins to the URDF file.

3. config: This folder will store the settings of the joint trajectory controller which will perform a trajectory motion by producing the execution motion plan for our robot arm to reach the goal position.

This package required below necessary ROS 2 packages which we can install via this linux's command:

```
sudo apt-get install
```

- ros-foxy-joint-state-broadcaster
- ros-foxy-joint-trajectory-controller
- Ros-foxy-controller-manager

3. **URDF:**

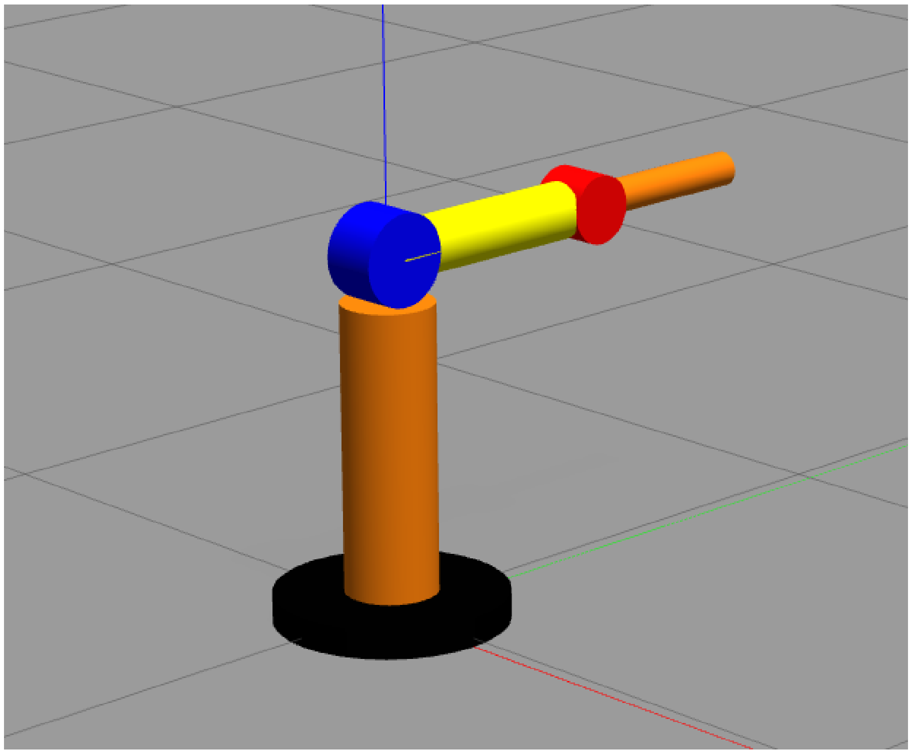
4. **Command:**

In order to run this application, after `colcon build`, we need to run these three commands:

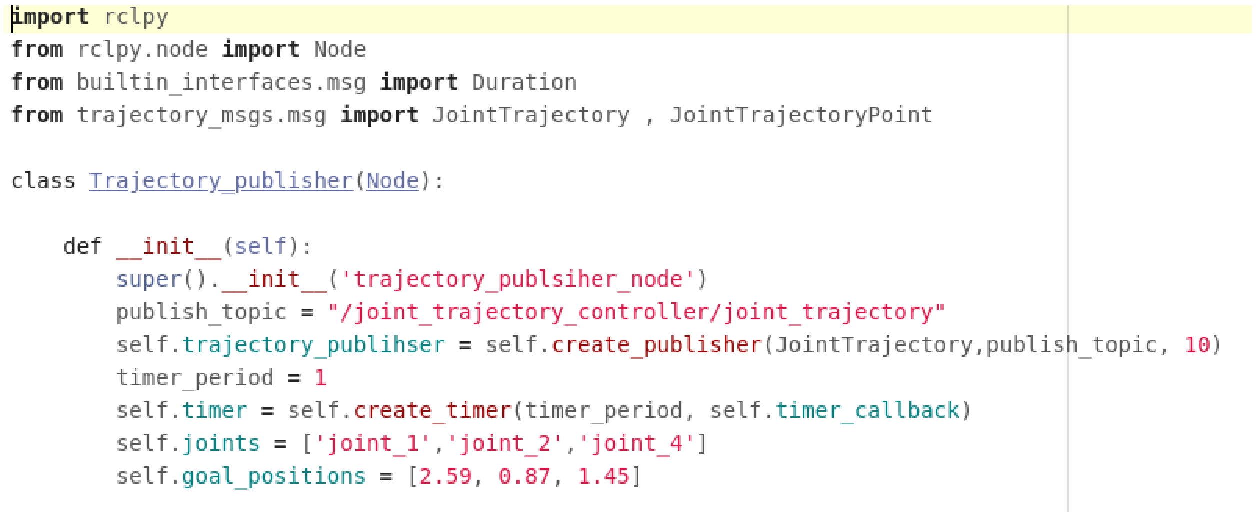
- source install/setup.bash
    - To source this package which makes it available to the ROS environment. After that, we need two terminals to run additional commands.
- ros2 launch arm controller.launch.py
    - This command is to open the Gazebo and spawn the robot arm based on the URDF file.
- ros2 run arm trajectory_test
    - This command is to send a trajectory of goal position to the robot arm.

5. **Outcome:**

When Gazebo is up and running and the robot arm has spawned to the world.



Below is a screenshot of the code that sends the goal position (2.59, 0.87, 1.45) to the robot arm.

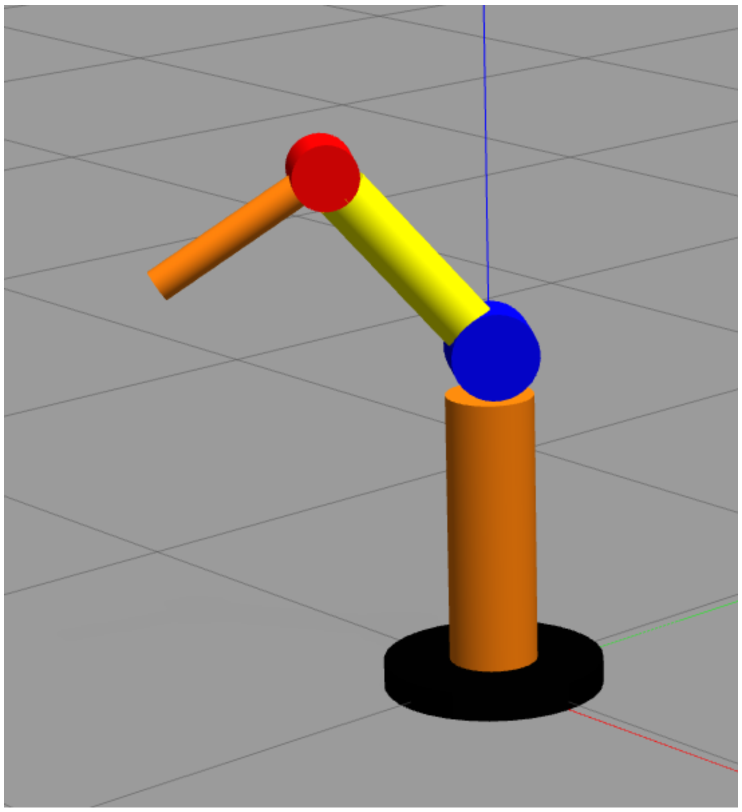


```
import rclpy
from rclpy.node import Node
from builtin_interfaces.msg import Duration
from trajectory_msgs.msg import JointTrajectory , JointTrajectoryPoint

class Trajectory_publisher(Node):

    def __init__(self):
        super().__init__('trajectory_publsiher_node')
        publish_topic = "/joint_trajectory_controller/joint_trajectory"
        self.trajectory_publihser = self.create_publisher(JointTrajectory,publish_topic, 10)
        timer_period = 1
        self.timer = self.create_timer(timer_period, self.timer_callback)
        self.joints = ['joint_1','joint_2','joint_4']
        self.goal_positions = [2.59, 0.87, 1.45]
```

Lastly, the robot arm will move to the position after I run the command to publish the trajectory node. (This command was added in a setup.py to be executable int the package)



## References:

[1]
https://docs.ros.org/en/foxy/Tutorials/URDF/Using-URDF-with-Robot-State-Publisher.html#publish-the-state

[2] https://automaticaddison.com/how-to-load-a-urdf-file-into-rviz-ros-2/

[3]
http://wiki.ros.org/urdf/Tutorials/Adding%20Physical%20and%20Collision%20Properties%20to%20a%20URDF%20Model

[4] http://gazebosim.org/tutorials?tut=ros_control&cat=connect_ros