Lab II: Basics of Robot Manipulator Control in Joint Space

Student: Dinmukhamet Murat 201917526

Task 1: Listener node for greater inputs

I wrote a node that continuously listens to incoming data, compares the new value with the last received value, and sends commands to the robot's joint if the new value exceeds the previous one. The purpose of this task was to ensure that the joint only moved in one direction, making the movement dependent on increasingly higher values. This node was successfully implemented and tested, showing the correct behavior where the robot's joint responded only to higher incoming values.

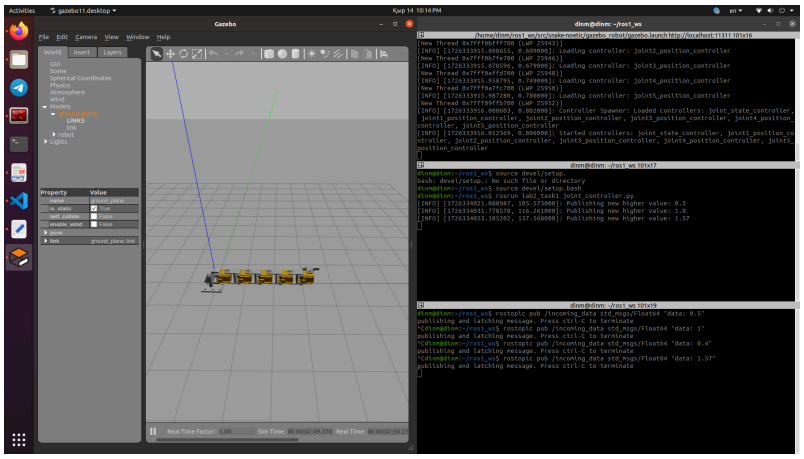

Fig-1. Output of task 1 with Gazebo.

Task 2: Square-Wave Step Response of Base and End-Effector Joints

A Python script (square_wave_pub.py) was created, which used ROS to publish square-wave position commands to both the base and end-effector joints. The square-wave alternated between 0 and 1.57 radians, simulating a step input. The robot joints responded by moving between 0 and 90 degrees as expected, with slight delays and imperfections in motion due to the dynamics of the system. However, after initial setup, there was a minor issue where the robot did not immediately respond on the first cycle but behaved as expected afterward.

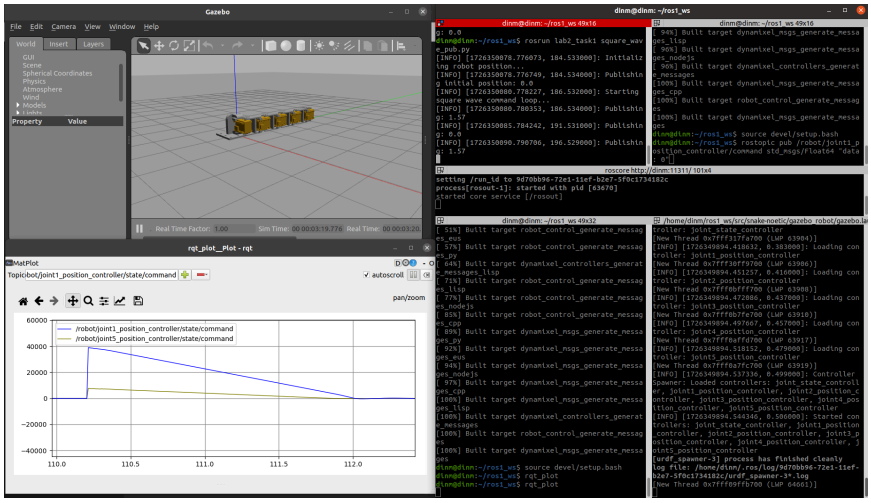

Fig-2. Gazebo and rqt plot outputs for task 2.

Task 3: Sine-Wave Response of Base and End-effector Joints

A new ROS node was created that publishes sine-wave values over time to both joints. This allows for a more fluid and continuous movement pattern, similar to how a real-world system might operate under varying inputs. The sine-wave function was implemented using the sin() function in Python, and the amplitude and frequency of the sine wave were adjusted to suit the robot's movement range. The robot moved as expected, with both the base and end-effector joints following the smooth oscillations of the sine wave.

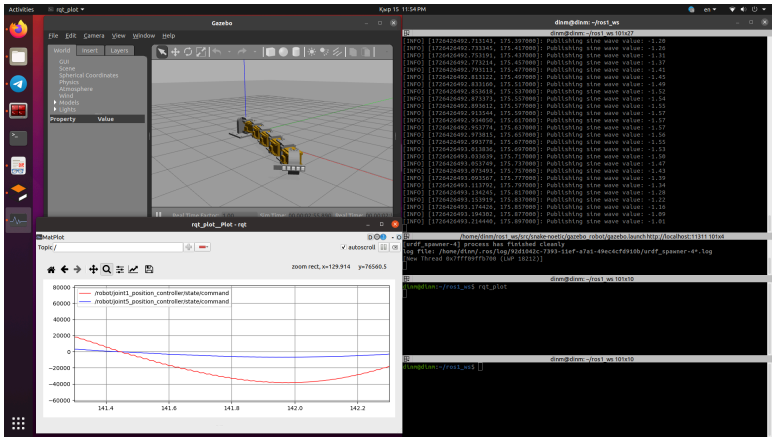

Fig-3. Gazebo and rqt plot outputs for task 3

Task 4: Tuning PID Gains

Using rqt's dynamic reconfigure tool, the P value was reduced for joints, which resulted in a noticeable change in the behavior of the robot. With the lowered P gain, the robot moved more sluggishly, which was expected since the Proportional term is responsible for the magnitude of the correction.

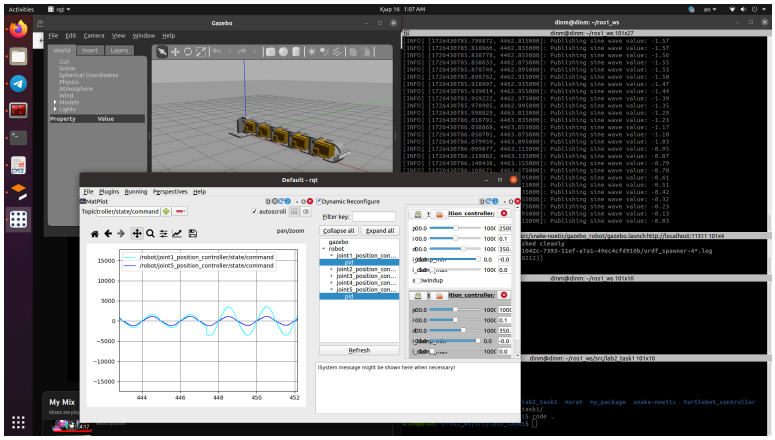

Fig-4. rqt plot output for task 4

Task 5: Snake movement

The final task was to make the robot move all its joints in the movement of a snake. Publishing sinusoidal commands with a phase shift for each joint created a wave-like motion of robot.

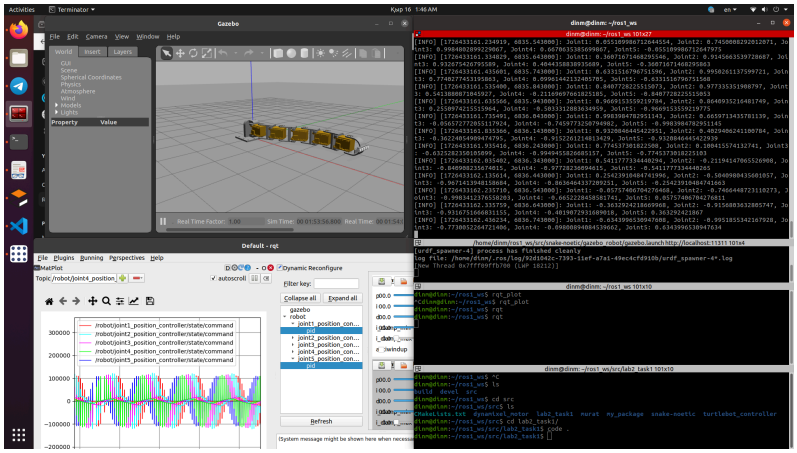

Fig-5. rqt plot with Gazebo output for task 5

Real Robot experiment:

Since the only way to run the real robot was using ROS Melodic, we changed the syntax of each task code, so there will be the same codes with addition _melodic in their names.

Also the video of snake motion will be attached in the github repository.

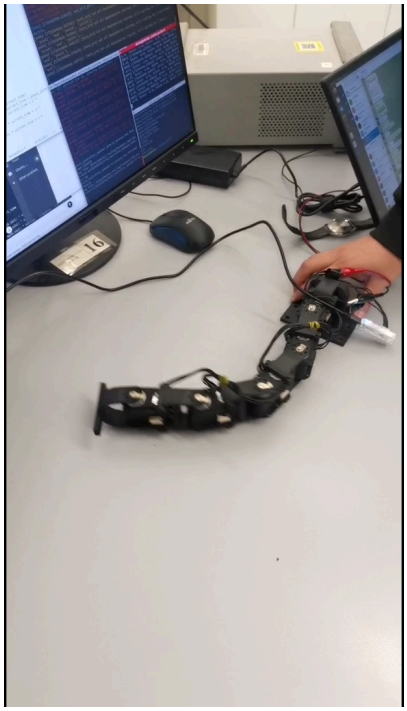

Fig-6. Real robot running