ROBT 403: Lab 3 - Joint-Space Trajectory Planning for a 3-DOF Planar Manipulator

Bauyrzhan Askarov
Nazarbayev University, SEDS, Robotics Engineering Undergraduate
Astana, Kazakhstan
bauyrzhan.askarov@nu.edu.kz

Dinmukhamet Murat
Nazarbayev University, SEDS, Robotics Engineering Undergraduate
Astana, Kazakhstan
dinmukhamet.murat@nu.edu.kz

I. Objectives

The primary objectives for this lab are as follows:

1) Derive the workspace for the planar 3-DOF robot.
2) Generate a cubic trajectory: We will use the 'cubic_coefficients' and 'cubic_trajectory' functions to calculate the polynomial coefficients and evaluate the joint trajectories at any time.
3) Implement the 'inverse_kinematics_with_orientation' function to compute the joint angles of the planar manipulator. The inputs are the target position and orientation for the end-effector.
4) Develop the 'plot_manipulator' function to visualize the manipulator's configuration during any point of the trajectory. This function will show the manipulator's joints, links, and indicate the end-effector's orientation.
5) Create a function to visualize joint-space trajectories between via points and illustrate the full end-effector path in Cartesian space.

II. Methodology

The tasks were implemented in Python code, and after obtaining the results, the code was ported to Gazebo for simulation using **Rospy**. The final step involved running the task on a physical robot equipped with a 3-DOF planar manipulator snake.

III. Task 1

The aim of this task is to compute the workspace for the 3-DOF planar robot shown in Fig. 1.

The "snake" robot has 5-DOF, but two of them were disabled as seen in Fig. 1. The joints are denoted by q and are numbered accordingly. The joint limits are set between [-90°, 90°] from the zero position.

Fig. 2 shows the workspace of the manipulator, which was computed by applying forward kinematics and varying the joint angles within their limits. The forwards kinematics are computed as indicated in figure3 where shows the calculation of the matrix. The code used to compute the workspace is included as a supplementary file.

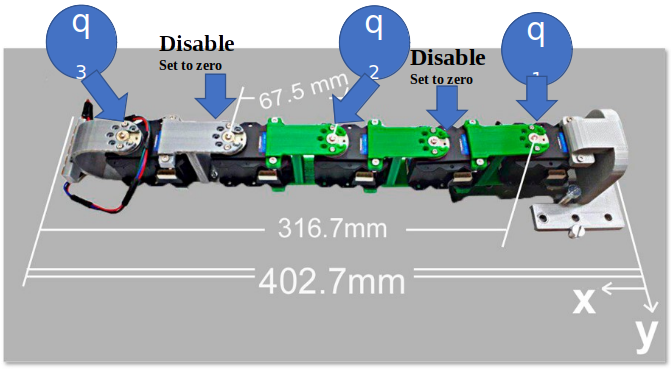

Fig. 1. Snake 3-DOF Planar Robot

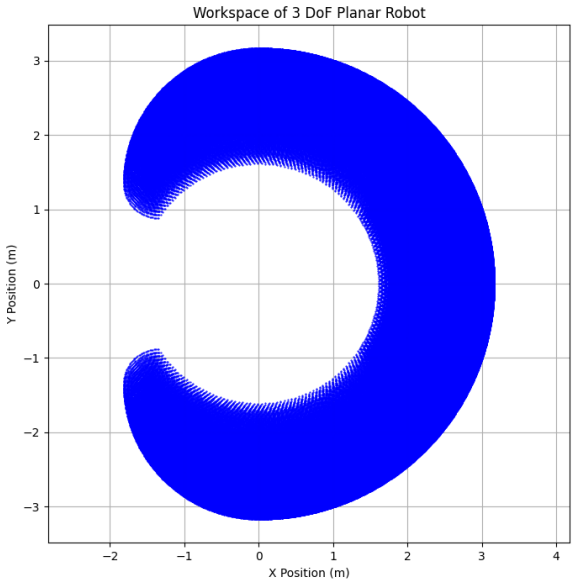

Fig. 2. Workspace of the 3-DOF Planar Robot

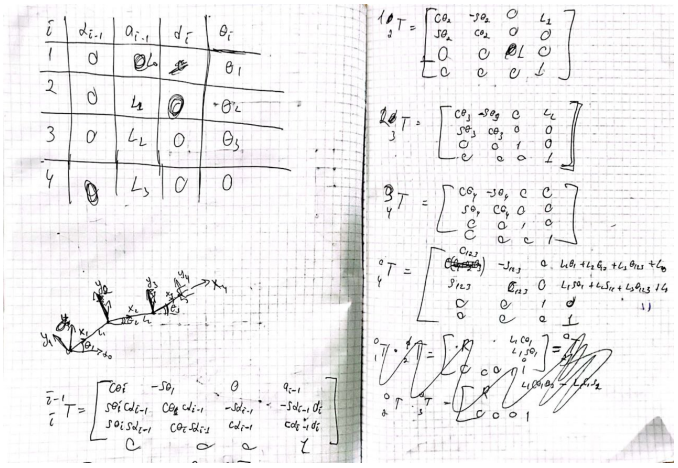

Fig. 3. Computation of the Forward Kinematics

IV. Task 2

The task's objective is to compute the cubic polynomial coefficients and evaluate the joint trajectories over time.

The cubic trajectory for joint angles is shown in Fig. 4, where joint q_1 moves from $\pi/2$ to $\pi/4$, q_2 from 0 to $\pi/4$, and q_3 from 0 to $-\pi/4$. The time interval used is [0, 10] seconds.

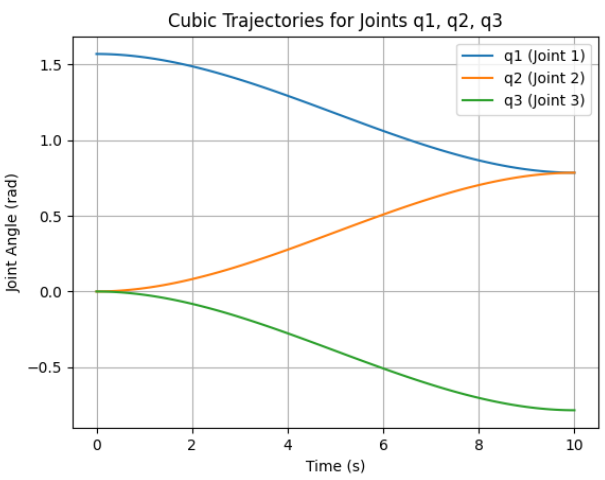

Fig. 4. 3 Cubic Joint Trajectory changes

V. Task 3

The goal of this task is to compute the joint angles for the planar manipulator using inverse kinematics. Inputs include the target x, y positions and orientation for the end-effector. The computation of the inverse kinematics can be seen in figure 5.

Fig. 6 displays the inverse kinematics results, showing the reachable joint angles for a target position of $X = 1$, $Y = 2$. Additional files, including the scripts and videos, are provided for review.

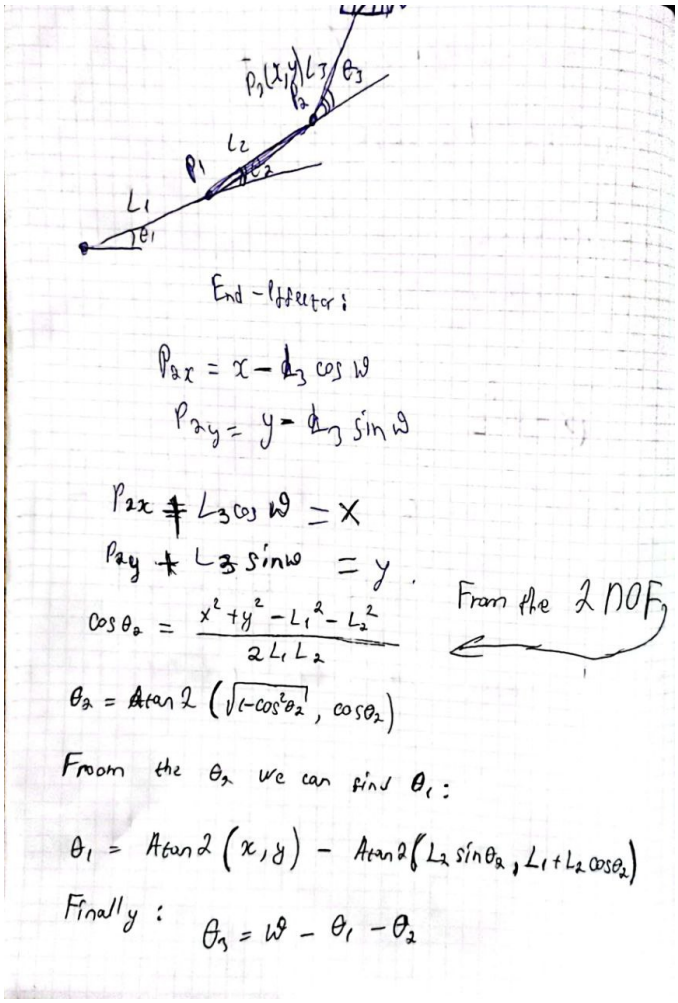

Fig. 5. Computation of IK for 3DOF

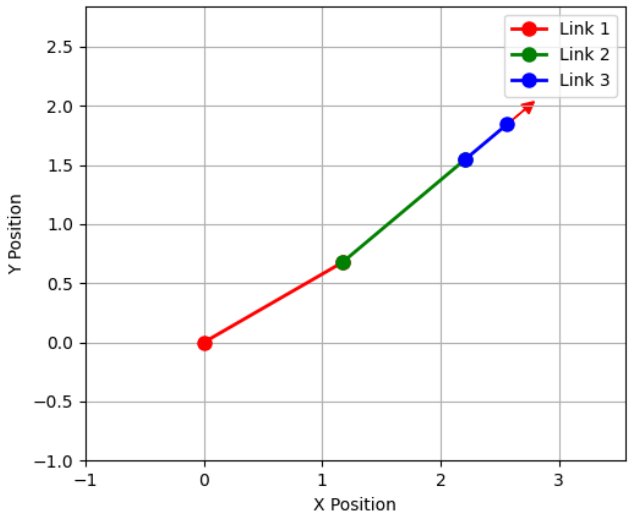

Fig. 6. Inverse Kinematics Output

VI. Task 4

This task involves plotting the manipulator's joints, links, and end-effector orientation for different configurations using forward kinematics.

Fig. 7 illustrates the 3-DOF planar manipulator in two positions. The links are color-coded, and the end-effector orientation is indicated by an arrow.

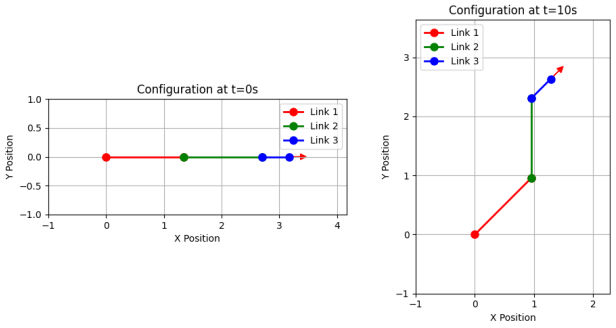

Fig. 7. Manipulator Configuration

VII. Task 5

The purpose of this task is to visualize the joint-space trajectories and generate the full end-effector path in Cartesian space.

The trajectory is visualized in Fig.8 . Additional scripts, graphs, and videos are provided as supplementary materials.

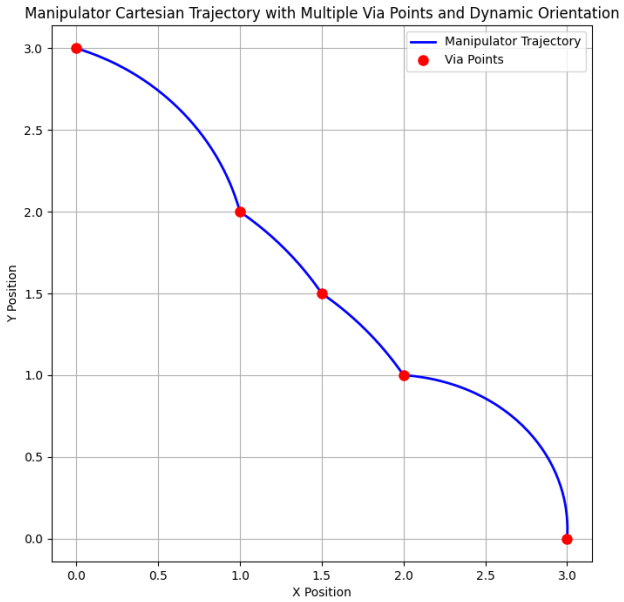

Fig. 8. Manipulator Path

VIII. Simulation and Real Robot Experiment

After running the tasks in python, the manipulator was simulated in Gazebo and then tested on the physical robot. The simulation results are similar to the real robot's behavior. Fig. 9 show the real-world experiment. Videos of experiments are available in our GitHub repository.

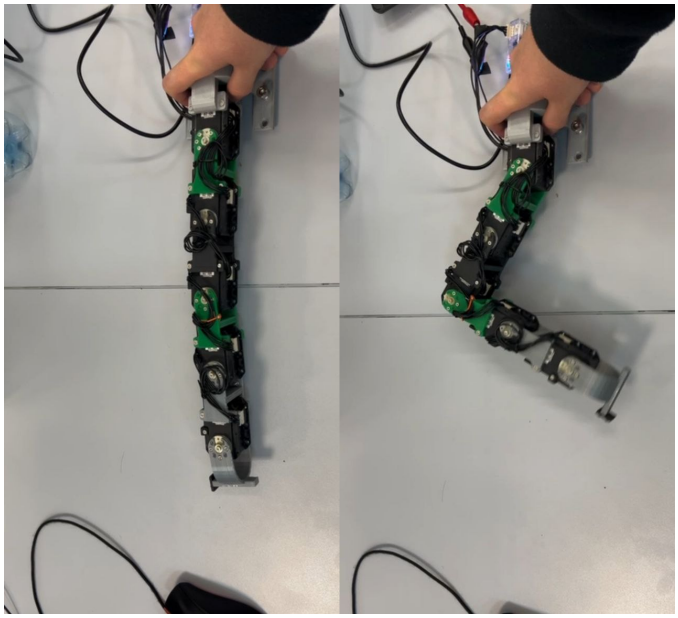

Fig. 9. Gazebo Simulation

IX. Discussion and Results

Referring to the task outputs, all the necessary work was completed. However, it's important to mention that some packages did not function properly and were not compatible with the current ROS setup. Specifically, it was necessary to use an earlier version of ROS, namely Melodic.

X. Conclusion

In conclusion, the laboratory assignment was successfully completed, and all tasks were finished. During the process, we worked on programming a 3DoF manipulator. Initially, we performed calculations for the workspace, applying knowledge from the lectures. Afterward, the forward and inverse kinematics of the manipulator were computed. Midway through, we calculated the cubic polynomial that defines the manipulator's trajectory. Additionally, the requirement to compute the via points was successfully fulfilled. All the code was written using Python and MATLAB scripts. The scripts were then uploaded to Gazebo to test the robot simulation. Finally, the real robot was tested, marking the successful completion of the assignment.