Exercise 1 Report: Implementation and Results

Heictor Costa

November 29, 2024

Introduction

This report documents my approach to solving the tasks outlined in Exercise 1, where I implemented kinematic modeling, inverse kinematics, and trajectory tracking for a robotic leg in the V-REP simulation environment. The primary goal was to accurately calculate the desired joint velocities to follow a specified trajectory and compare the desired and actual trajectories.

Implementation

The tasks required me to compute the relative rotation matrices, homogeneous transformation matrices, Jacobians, and solve inverse kinematics using the Newton-Raphson method and inverse differential kinematics. Below, I describe my approach to each task in detail:

Task 1: Relative Rotation Matrices

I derived the relative rotation matrices using symbolic definitions for joint angles α, β, and γ. Each matrix corresponded to a rotation about one of the axes of the robotic leg's coordinate frames.

Task 2: Homogeneous Transformation Matrices

Using the rotation matrices, I calculated the homogeneous transformation matrices for each joint. These transformations were combined to compute the cumulative transformation matrix H_{B3}, which describes the position and orientation of the end-effector relative to the base frame.

Task 3: Jacobian Calculation

I implemented the Jacobian J_{BF}^{B} symbolically by taking the partial derivatives of the end-effector position r_{BF}^{B} with respect to the generalized coordinates $[\alpha, \beta, \gamma]^T$. This Jacobian allowed me to compute joint velocities for a desired Cartesian velocity.

Task 4: Inverse Kinematics

To solve the inverse kinematics problem, I used the Newton-Raphson method. Starting with an initial guess for the joint angles, I iteratively updated the angles by minimizing the error between the desired and current positions of the end-effector. This method ensured convergence to the desired position $r_{Goal} = [0.2, 0.5, -2]^T$.

Task 5: Trajectory Tracking

I implemented inverse differential kinematics to track a circular trajectory in the body plane. Using a proportional controller, I computed the desired Cartesian velocity v to minimize the error between the current and desired positions. The Jacobian pseudo-inverse was then used to compute the joint velocities needed to follow the trajectory.

Results

The robotic leg successfully followed the desired circular trajectory in the simulation. The plot below compares the desired and obtained trajectories. The desired trajectory is represented by a red dashed line, and the actual trajectory is represented by a blue solid line.

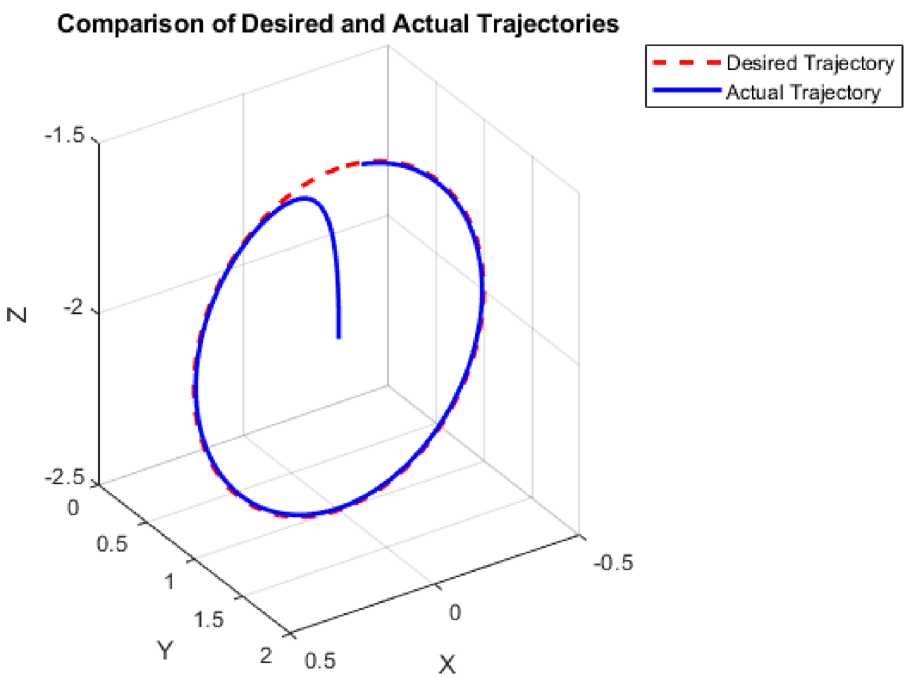

Figure 1: Comparison of Desired and Actual Trajectories

Discussion

The results indicate that the robotic leg was able to closely follow the desired trajectory. The use of the proportional controller effectively minimized the error between the desired and actual positions. However, minor deviations were observed, particularly at the start of the trajectory. These deviations can be attributed to:

- The initial guess for the joint angles, which may have been far from the ideal configuration.
- The limited accuracy of the Jacobian pseudo-inverse in handling singularities.
- Delays in simulation updates due to the time resolution Δt.

To improve accuracy, I could explore adaptive control gains and higher-order inverse kinematics methods. Additionally, increasing the simulation's time resolution might reduce discrepancies in real-time trajectory tracking.

Conclusion

Through symbolic modeling, inverse kinematics, and trajectory tracking, I successfully solved the tasks outlined in Exercise 1. The comparison of the desired and obtained trajectories demonstrates that the implemented algorithms are effective in controlling the robotic leg. Future work could focus on optimizing control strategies and enhancing simulation precision.