Assignment 2 (Hands-on): Jacobians

Robot Kinematics and Dynamics

Prof. Jeff Ichnowski
Shahram Najam Syed
Yuemin Mao

Contents

1 Hands-on

1. **Implementing and Testing the Jacobian Calculation** You are provided with a script named `jacobian_RR.py`. In this script, you need to implement the Jacobian calculation for a 2D RR (Revolute-Revolute) robot arm. Your task is to complete the `calculate_jacobian_RR` function:

   ```
   def calculate_jacobian_RR(fa, L1=0.316, L2=0.384+0.088):
       # Get current joint angles
       joints = fa.get_joints()
       theta1 = -joints[1]  # Negating to match the coordinate system
       theta2 = joints[3]

       # Calculate reduced 2x2 Jacobian for planar motion
       J_reduced = np.zeros((2, 2))
       J_reduced[0, 0] = ...
       J_reduced[0, 1] = ...
       J_reduced[1, 0] = ...
       J_reduced[1, 1] = ...

       return J_reduced
   ```

   Fill in the ellipses (...) with the correct Jacobian calculations. Use the provided joint angles (theta1 and theta2) and link lengths (L1 and L2) in your calculations.

   Once you have implemented the function, run the `jacobian_RR.py` script. This script will:

   - Move the robot to an initial position.
   - Execute a pre-recorded trajectory.
   - Calculate the Jacobian at each timestep using your implementation.
   - Compare your calculated Jacobian with the one obtained from the robot's API.
   - Save the differences between the Jacobians.

2. **Deliverables**
   - The completed `jacobian_RR.py` script with your implementation of the `calculate_jacobian_RI` function.
   - The `jacobian_differences.npy` file generated by the script, which contains the differences between your calculated Jacobian and the API Jacobian at each timestep.

Note: This exercise focuses on joints 1 and 3 of the Franka arm to simulate a 2D RR arm configuration. The provided script handles the robot movement and data collection, allowing you to focus on implementing and understanding the Jacobian calculation. The comparison between your calculated Jacobian and the API Jacobian will help you validate your implementation and gain insights into the practical aspects of robot kinematics.

1) Submission

To submit, run `create_submission_hands_on.py`. It will first check that your submitted files run without error, and perform a small sanity check. Note, this is not going to grade your submission! The function will create a file called `<andrew_id>.zip`.

2 Submission Checklist

- ☐ Create a PDF of your answers to the written questions with the name `<andrew_id>.pdf` and submit it to Gradescope.
- ☐ Run `create_submission.py` in the python terminal.
- ☐ Upload `<andrew_id>.zip` to Gradescope.
- ☐ Coding questions are autograded, with no limit on number of submissions within the mandated deadline.