Report Homework 2

Slava Koshman

December 2024

Problem 1

Design the outer loop as described above using the logarithmic map for orientation error. Provide and explain the complete set of equations describing your controller.

1. System Dynamics

The dynamics of the robotic manipulator can be expressed as:

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \tau + J(q)^T f \tag{1}$$

where: q = the joint position vector.
$M(q)$ = the mass/inertia matrix.
$C(q,\dot{q})$ = the Coriolis/centrifugal matrix.
$g(q)$ = represents gravitational torques.
τ = the vector of joint torques.
$J(q)$ = the Jacobian matrix mapping joint velocities to task-space velocities.
f = the generalized force at the end-effector.

2. Task-Space Dynamics

The task-space acceleration is related to the joint-space variables by:

$$\ddot{x} = J(q)\ddot{q} + \dot{J}(q,\dot{q})\dot{q} \tag{2}$$

Rewriting the system for task-space control:

$$M_x\ddot{x} + C_x\dot{x} + g_x = f \tag{3}$$

where: $M_x = (\mathrm{J\ M}^{-1}J^T)^{-1}$
$C_x = \mathrm{M}_x JM^{-1}C - M_x\dot{J}$
$g_x = \mathrm{M}_x JM^{-1}g$

3. Desired End-Effector Pose Tracking

The desired end-effector pose consists of:

- **Position**: x_d (translation in 3D space),
- **Orientation**: R_d (desired rotation matrix).

Define the pose error:

1. **Position Error**:

$$e_p = x_d - x \tag{4}$$

2. **Orientation Error**: Use the logarithmic map:

$$e_o = \log\big(R_d R^T\big) \tag{5}$$

where R is the current orientation matrix, and $\log(R)$ maps the rotational matrix to its axis-angle representation (a vector in $\mathbb{R}^3$).

4. Outer Loop Control Law

Combine position and orientation errors into a single task-space error:

$$e = \begin{bmatrix} e_p \\ e_o \end{bmatrix} \tag{6}$$

The desired task-space acceleration is:

$$\ddot{x}_d = K_p e + K_d \dot{e} - \dot{J}\dot{q} \tag{7}$$

where: K_p = Diagonal proportional gain matrix
K_d = Diagonal derivative gain matrix

Substitute this into the task-space dynamics:

$$f = M_x \ddot{x}_d + C_x \dot{x} + g_x \tag{8}$$

5. Joint-Space Torque

To compute the joint torques, map the task-space force back using the Jacobian:

$$\tau = J^T f, \tag{9}$$

6. Complete Controller Summary

The TSID controller includes the following steps:

1. **Error Computation**:

$$e_p = x_d - x \tag{10}$$

$$e_o = \log(R_d R^T) \tag{11}$$

$$e = \begin{bmatrix} e_p \\ e_o \end{bmatrix} \tag{12}$$

2. **Desired Task-Space Acceleration**:

$$\ddot{x}_d = K_p e + K_d \dot{e} - \dot{J}\dot{q} \tag{13}$$

3. **Task-Space Force**:

$$f = M_x \ddot{x}_d + C_x \dot{x} + g_x \tag{14}$$

4. **Joint-Space Torque**:

$$\tau = J^T f \tag{15}$$

Problem 2

Implement posture regulation using modeling tools like Pinocchio for computing dynamics, kinematics, and Jacobians. Implement your controller in the MuJoCo simulator and provide plots demonstrating tracking convergence and control signals. Robot descriptions (URDF and XML) are provided. A template with a movable target site is available for your convenience.

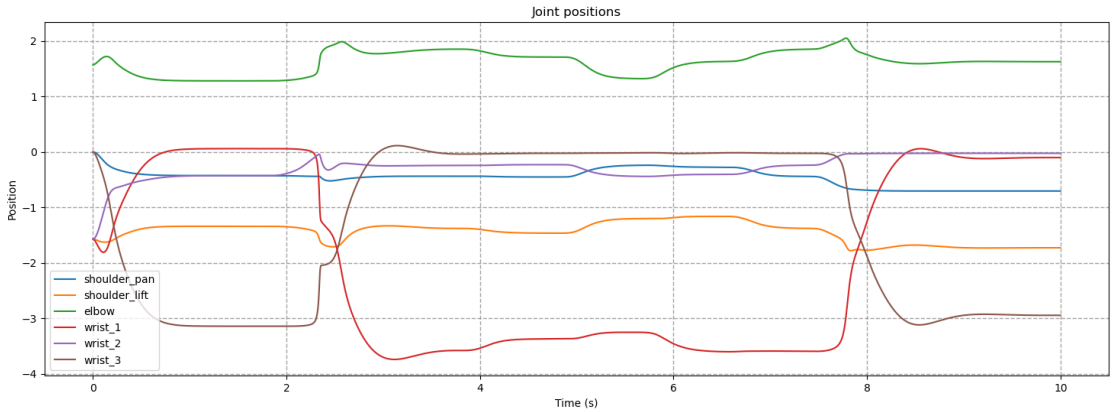

Figure 1: Plot illustrating the joint positions while the end-effector follow to target

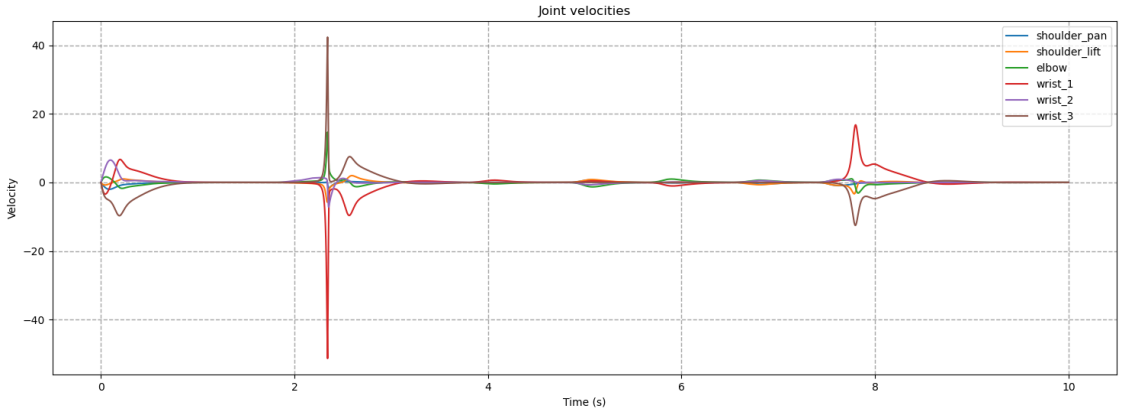

Figure 2: Plot illustrating the joint velocities while the end-effector follow to target

Problem 3

Demonstrate your algorithm by making the robot follow a task space trajectory (e.g., a circular path while maintaining the end-effector orientation toward the center).

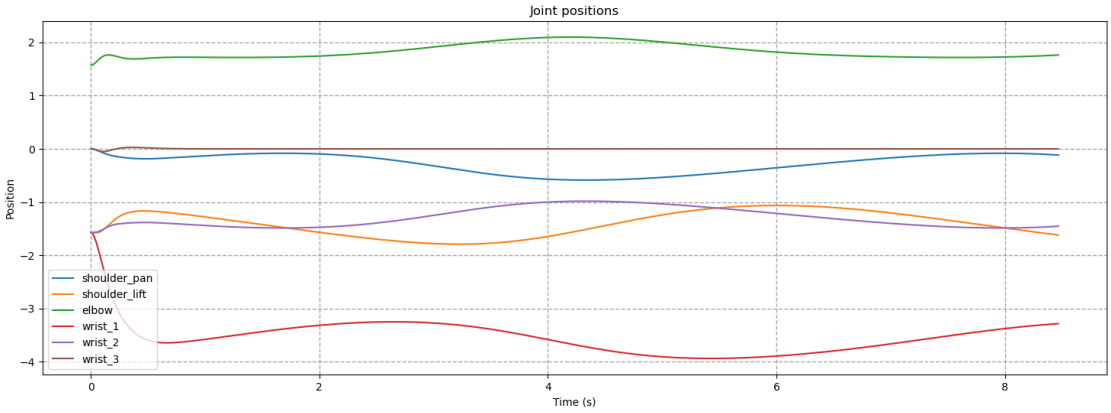

Figure 3: Plot illustrating the joint positions while the end-effector moving by circler trajectory

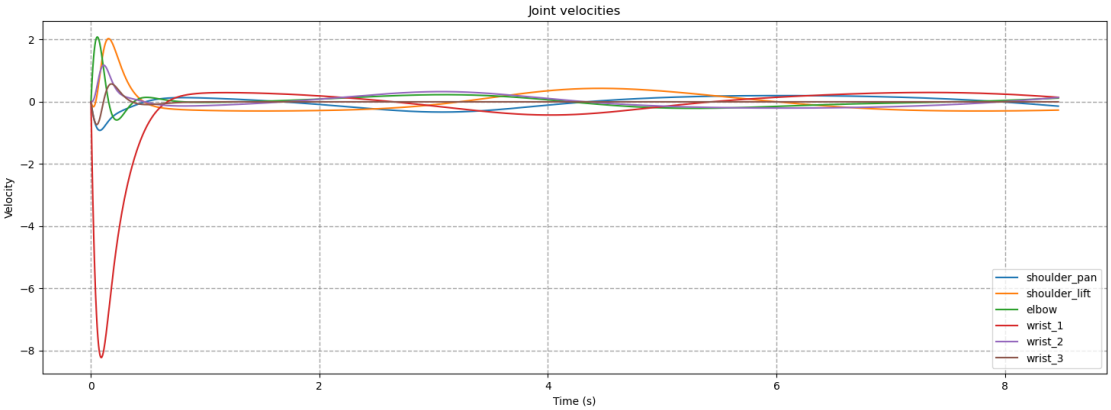

Figure 4: Plot illustrating the joint velocities while the end-effector moving by circler trajectory