Indian Institute of Technology, Gandhinagar

ME 691-XI

Advanced Robotics

Semester–I, Academic Year 2023-24

Class Project- 2
FRANKA EMIKA PANDA 7-DoF ROBOT

By

Team: *AK 47*

Anavart Pandya 20110016

Kush Patel 20110131

1 INTRODUCTION

In this report, we discuss the formulation of dynamics for the Panda robot, a 7-degree-of-freedom (7-DOF) robotic manipulator. The dynamics formulation is crucial for understanding and controlling the robot's motion, considering factors such as inertia, Coriolis effects, gravity, and friction. We will follow a standardized approach to derive the equations of motion.

Figure 1: Franka Emika Panda 7-Dof Robot

1.1 Preliminary Resources

Before delving into dynamics formulation, let's acknowledge the resources available, particularly the GitHub repository providing essential dynamics parameters such as the inertia matrix (D), Coriolis matrix (C), Coriolis vector (c), gravity vector (g), and friction torque matrix (Fr).

2 DYNAMICS SIMULATION

Step 1: Define Joint Space Trajectory
We initiate by defining a joint space trajectory. Utilizing the inverse dynamics approach, we calculate the joint torques required to follow this trajectory. A numerical solver is employed for accurate results.

Step 2: Verify Inverse Dynamics
To validate the inverse dynamics formulation, we compare the desired joint space trajectory with the actual trajectory obtained through numerical integration. The overlapping of these trajectories indicates the accuracy of the inverse dynamics approach.

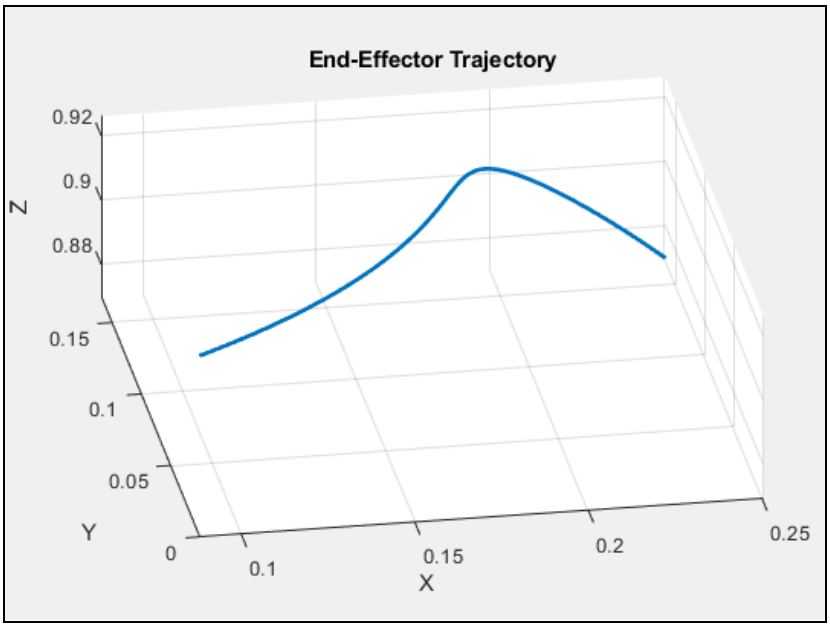

Figure 2: Task Space Trajectory when ($q_{desired} = sin(t)$) [Without Control]

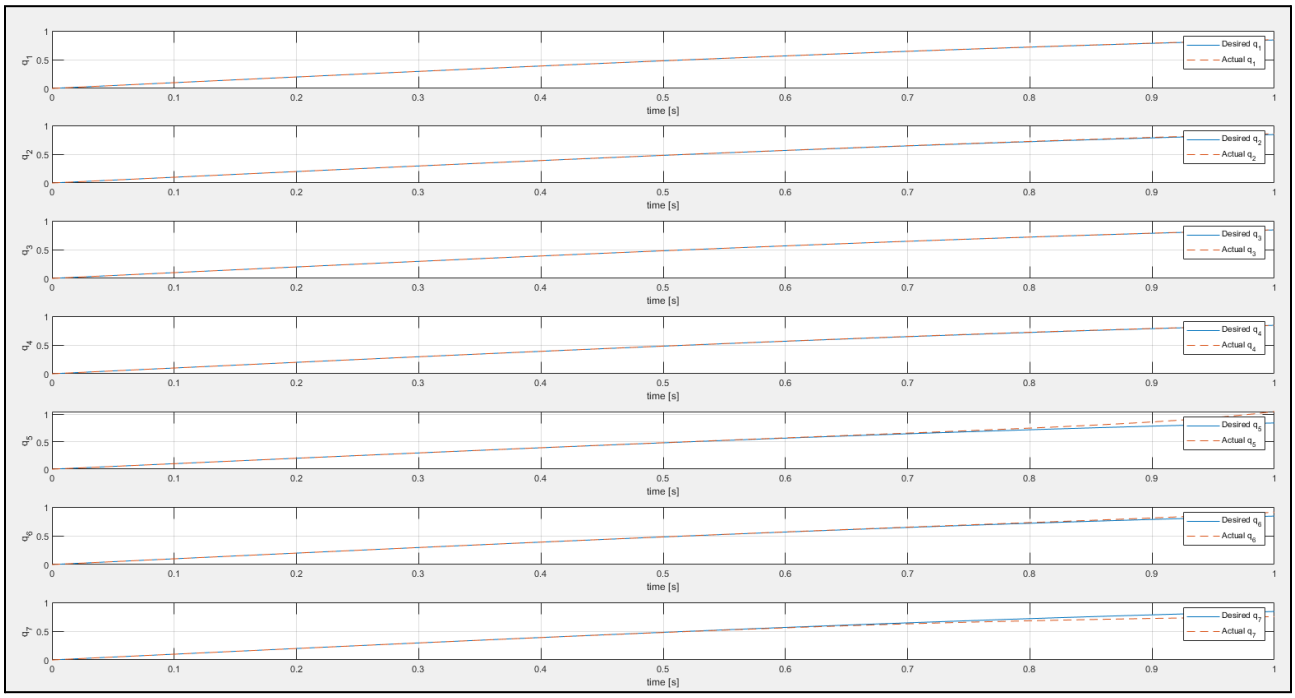

Figure 3: Joint Space Trajectory ($q_{desired} = sin(t)$) [Without Control]

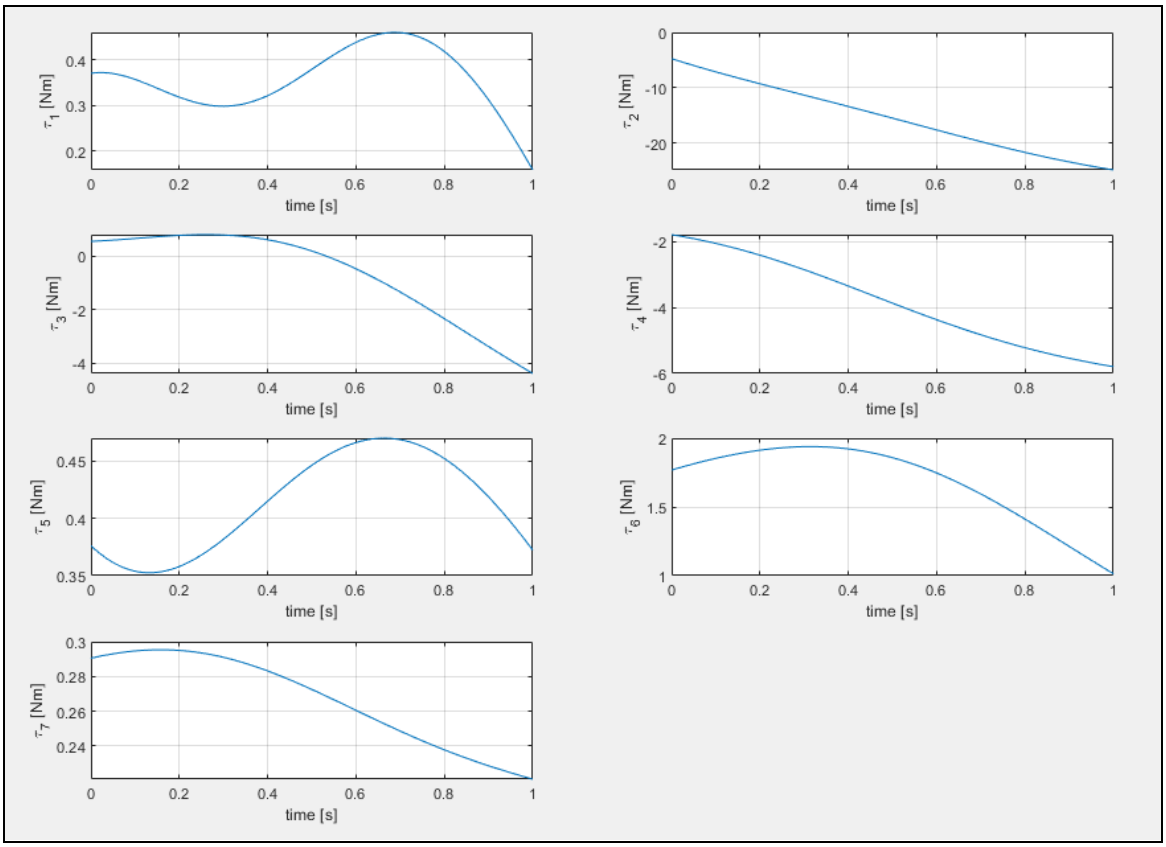

Figure 4: Joint Torques [Without Control]

Step 3: Implement PD Control

Integrating Proportional-Derivative (PD) control within inverse dynamics enhances the efficiency of trajectory tracking. This involves tuning the control gains (Kp and Kd) to achieve desired performance.

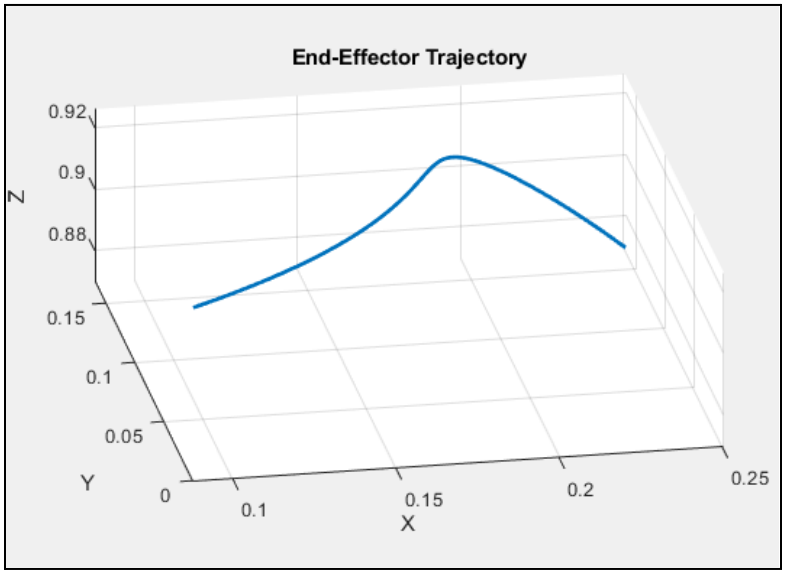

Figure 5: Task Space Trajectory when ($q_{desired} = sin(t)$) [With PD Control]

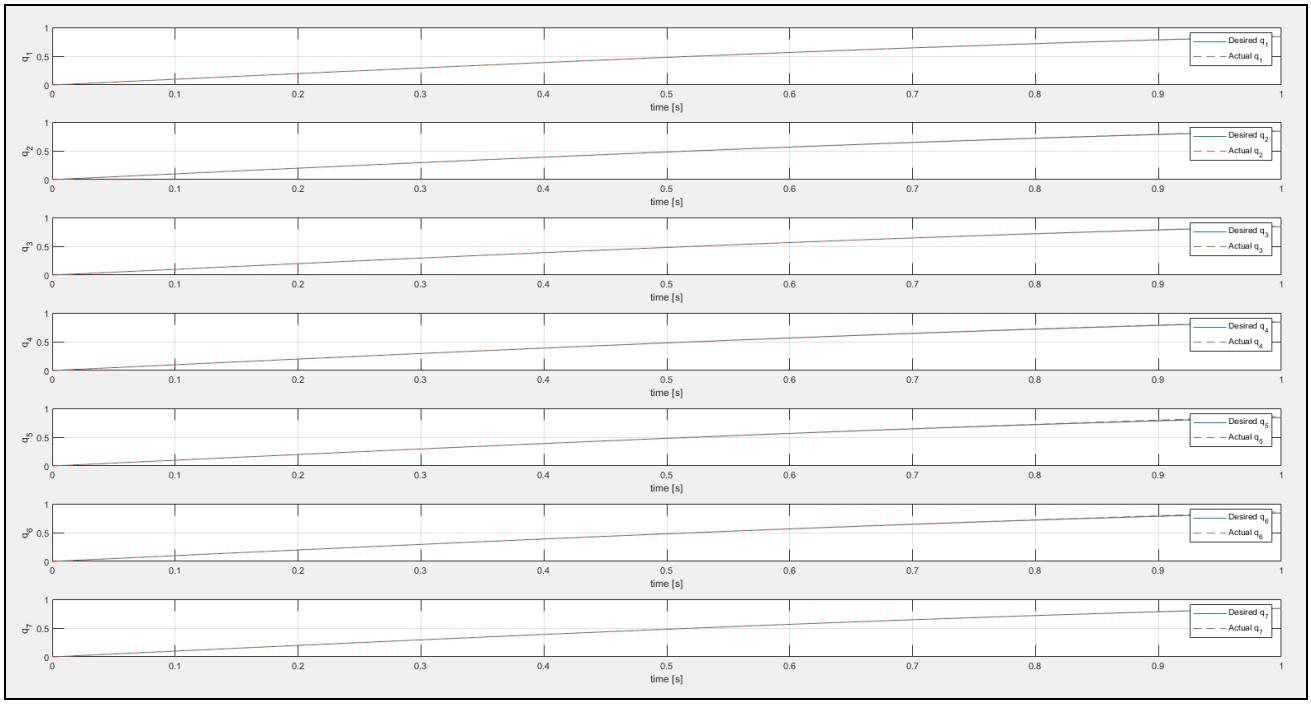

Figure 6: Joint Space Trajectory $(q_{desired} = sin(t))$ [With PD Control]

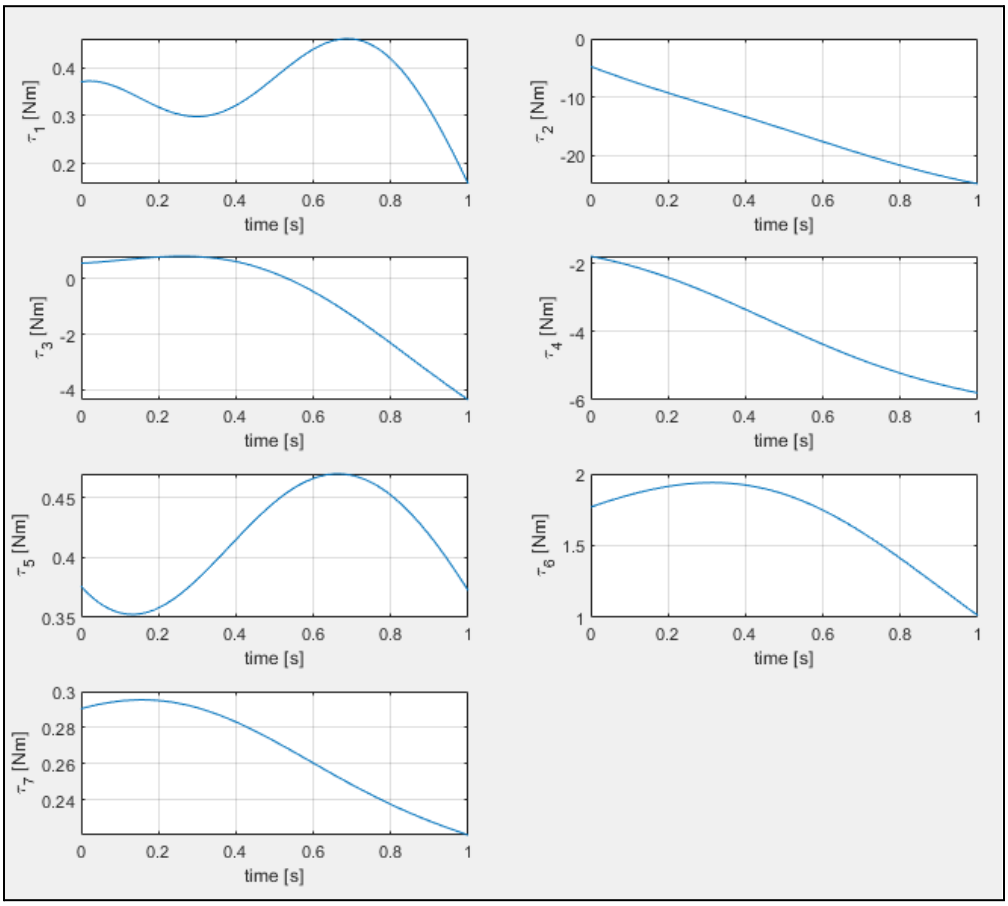

Figure 7: Joint Torques [With PDControl]

3 TRAJECTORY GENERATION:

3.1 Straight-Line Trajectory:-

Let's consider a straight-line trajectory in the Cartesian space between two points (x0,y0) and (xf,yf). We considered the z dimensional value to be constant and orientation is also constant. The trajectory can be represented as a cubic polynomial in both x and y directions as follows:

$$x(t) = a_1 + a_2 t + a_3 t^2 + a_3 t^3$$
$$y(t) = b_1 + b_2 t + b_3 t^2 + b_4 t^3$$

To obtain these coefficients, we need to consider the initial and final conditions of the trajectory. For a straight-line trajectory, the initial and final conditions are specified as follows:

$$At\, t = 0 :\ x(0) = x_0,\ y(0) = y_0,\ \dot{x}(0) = 0,\ \dot{y}(0) = 0$$
$$At\, t = t_f :\ x(t_f) = x_f,\ y(t_f) = y_f,\ \dot{x}(t_f) = 0,\ \dot{y}(t_f) = 0$$

By imposing these conditions, we can solve for the coefficients. The resulting trajectory equations will ensure that the end-effector starts from rest at the initial point and comes to rest again at the final point.

4 REDUNDANCY RESOLUTION:

4.1 Case I: Minimum Norm Solution:-

For this case, we consider a 6x7 Jacobian (J) as we are not interested in the rotational part. The desired joint velocities (q dot_desired) are calculated using the inverse of the Jacobian, and the minimum norm solution is employed.

$$\dot{q}_{desired} = \left(J^{+}\dot{X}_{desired}\right) + \left(I - J^{+}J\right)w$$

Where $J^{+}$is the pseudoinverse of J. The term $\left(I - J^{+}J\right)w$represents the null space optimization, which can be set to zero if no specific preference is given.

4.2 Case II: Minimize Joint Velocity Norm:-

Similar to Case II, we have a 6x7 Jacobian (J). Redundancy resolution involves minimizing the joint velocity norm using the Gradient Projection Method. The optimization is performed by assuming $w = -K_h \delta(H)$, where H is the joint velocity norm, and δ(H) is the gradient of H.

$$\dot{q}_{desired} = \left(J^{+}\dot{X}_{desired}\right) + \left(I - J^{+}J\right)w$$

The gradient of H is $\begin{bmatrix}\dot{q_1} & \dot{q_2} & \dot{q_3}\end{bmatrix}^T$ The term Kh is a scaling factor. The result is an optimized joint velocity minimizing the norm.

Redundancy resolution involves adapting the Jacobian matrix to the specific requirements of each case, and optimization is performed to achieve desired characteristics such as minimum norm solution, minimized joint velocity norm, or optimized velocity manipulability. The choice of optimization strategy depends on the specific objectives and constraints of the robotic system.

5 DYNAMICS MOTION CONTROL

Dynamic motion control involves the use of control strategies to regulate the motion of a robotic system by considering its dynamic characteristics. In the context of Panda robot, dynamic motion control typically encompasses trajectory tracking, where the goal is to make the end-effector follow a desired trajectory accurately. The process involves both inverse dynamics and forward dynamics, and control strategies are employed to ensure accurate tracking.

5.1 Inverse Dynamics:

Inverse dynamics is the process of determining the joint torques or forces required to achieve a desired end-effector motion. For panda robot, the inverse dynamics equation is given by:

$$\tau = D(q)\ddot{q} + C(q,\dot{q})\dot{q} + \phi(q) + Fr + \tau_{ext}$$

Where: τ=Joint torque vector

$D(q)$=Inertia matrix

$C(q,\dot{q})$=Christoffel matrix

$\phi(q)$= Gravity vector

Fr = Friction vector

$$D(q) = m_1 J_{V_{c_1}}^T J_{V_{c_1}} + m_2 J_{V_{c_2}}^T J_{V_{c_2}} + I_{1_{mask}} + I_{1_{mask}} + I_{2_{mask}} + I_{4_{mask}} + I_{5_{mask}} + I_{6_{mask}} + I_{7_{mask}}$$

we can compute the Christoffel symbols using the following equation:

$$c_{ijk} = \frac{1}{2}\left\{ \frac{\partial d_{kj}}{\partial q_i} + \frac{\partial d_{ki}}{\partial q_j} - \frac{\partial d_{ij}}{\partial q_k} \right\}$$

Note that, for a fixed k, we have $c_{ijk} = c_{jik}$, which reduces the effort involved in computing these symbols by a factor of about one half.

For defining phi:

$$\phi_k = \frac{\partial V}{\partial q_k}$$

then we can write the Euler-Lagrange equations as:

$$\Sigma\left[d_{kj}(q)\ddot{q}_j\right] + \Sigma\left[c_{ijk}(q)\dot{q}_i\dot{q}_j\right] + \phi_k(q) = \tau_k \ , \ k = 1,2,3,4,5,6,7$$

The inverse dynamics process is crucial for motion control, as it provides the required joint torques to achieve a specified trajectory.

5.2 Forward Dynamics:

Forward dynamics involves predicting the motion of the manipulator given the joint torques. The numerical integration approach used in the code snippet you provided performs this task. The forward dynamics equation can be expressed as:

$$\ddot{q} = D(q)^{-1}[\tau - (C(q,\dot{q})\dot{q} + \phi(q) + Fr + \tau_{ext})]$$

The numerical integration loop in the code calculates the joint accelerations by solving the above equation at each time step. The integration proceeds to update joint velocities and positions

5.3 Control Strategies:

Control strategies are implemented to enhance trajectory tracking and improve the manipulator's performance. The snippet uses a proportional-derivative (PD) controller for feedback control. The control law is given by:

$$\tau_{Feedback} = K_p q_{error} + K_d \dot{q}_{error}$$

Where Kp,Kd = Control gains

q_{error} = Difference between the desired and actual joint positions

$\dot{q}_{error}$ = Difference between the desired and actual joint velocities

The total joint torque (τ) is then composed of the feedforward torque obtained from the inverse dynamics and the feedback torque from the PD controller:

$$\tau = \tau_{Feedforward} + \tau_{feedback}$$

This control input is used in the forward dynamics calculation to update the joint accelerations, velocities, and positions.

Dynamic motion control, as implemented in the provided code, combines inverse dynamics, forward dynamics, and feedback control to ensure accurate trajectory tracking for the Panda robot. The use of numerical integration allows for the simulation of the manipulator's motion over time, considering its dynamic interactions with the environment. The PD controller adds a feedback loop to mitigate tracking errors and improve the overall performance of the system.

6 RESULTS & ANALYSIS:

Plots for task space trajectories, joint space trajectories, and comparison in joint angle trajectories are generated for each case with control & without control. The two cases are analyzed in terms of the manipulator's performance, considering factors such as trajectory tracking accuracy, joint space redundancy resolution, and computational efficiency.

The focus is on evaluating the effectiveness of different redundancy resolution strategies and the impact of control mechanisms on task space and joint space trajectories.

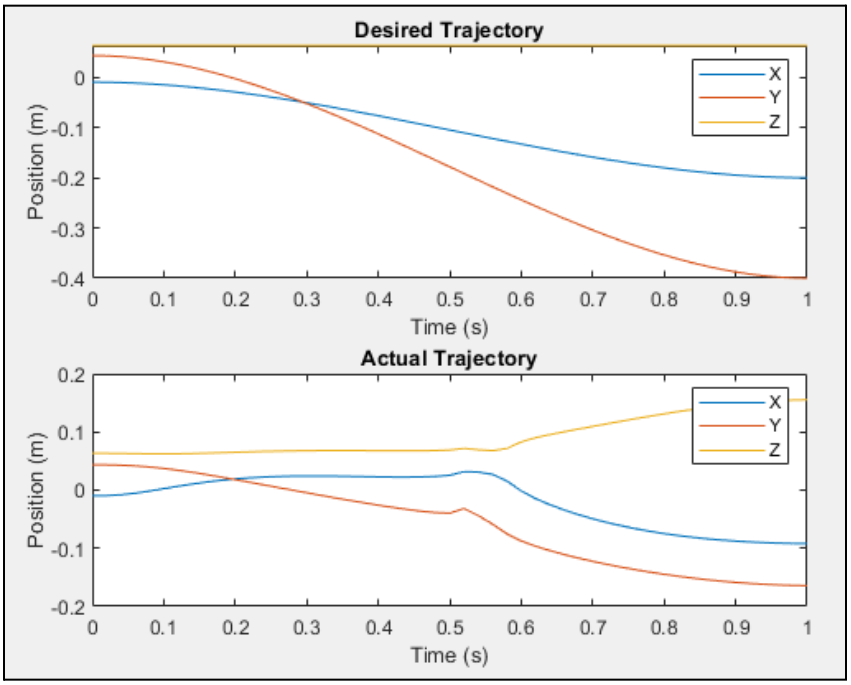

Figure 8: Task Space Trajectory (Straight Line)

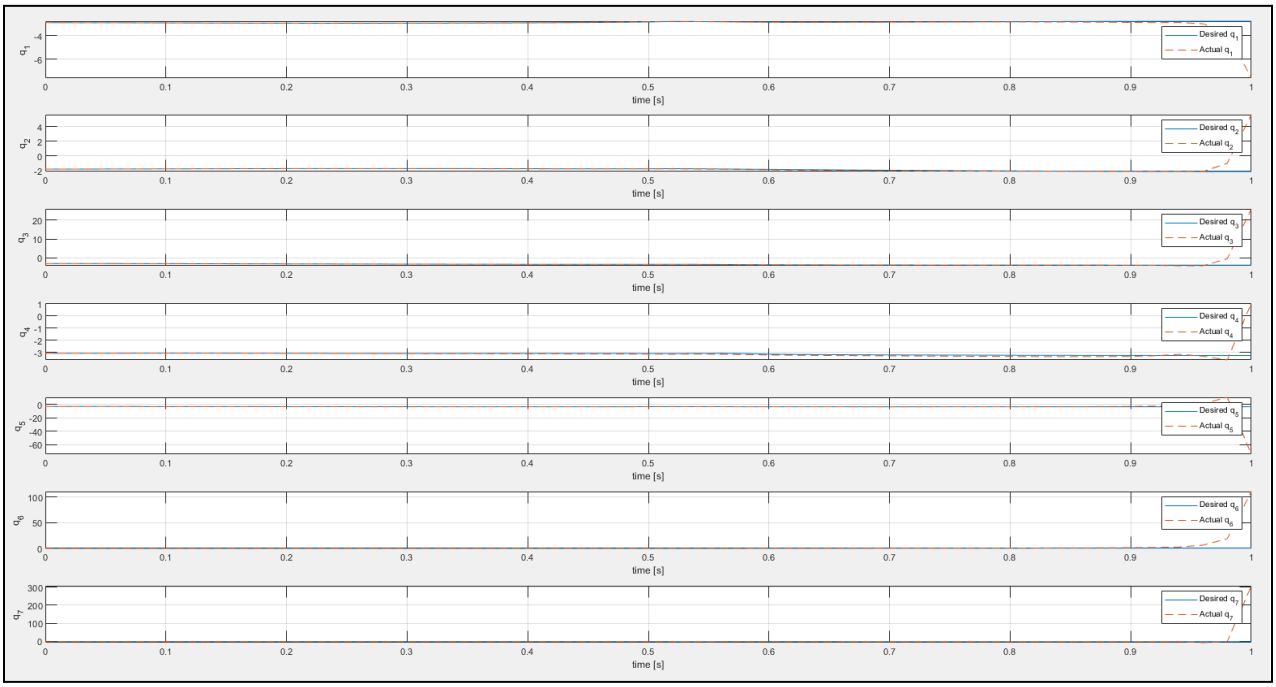

Figure 9: JointSpace Trajectory (Straight Line)

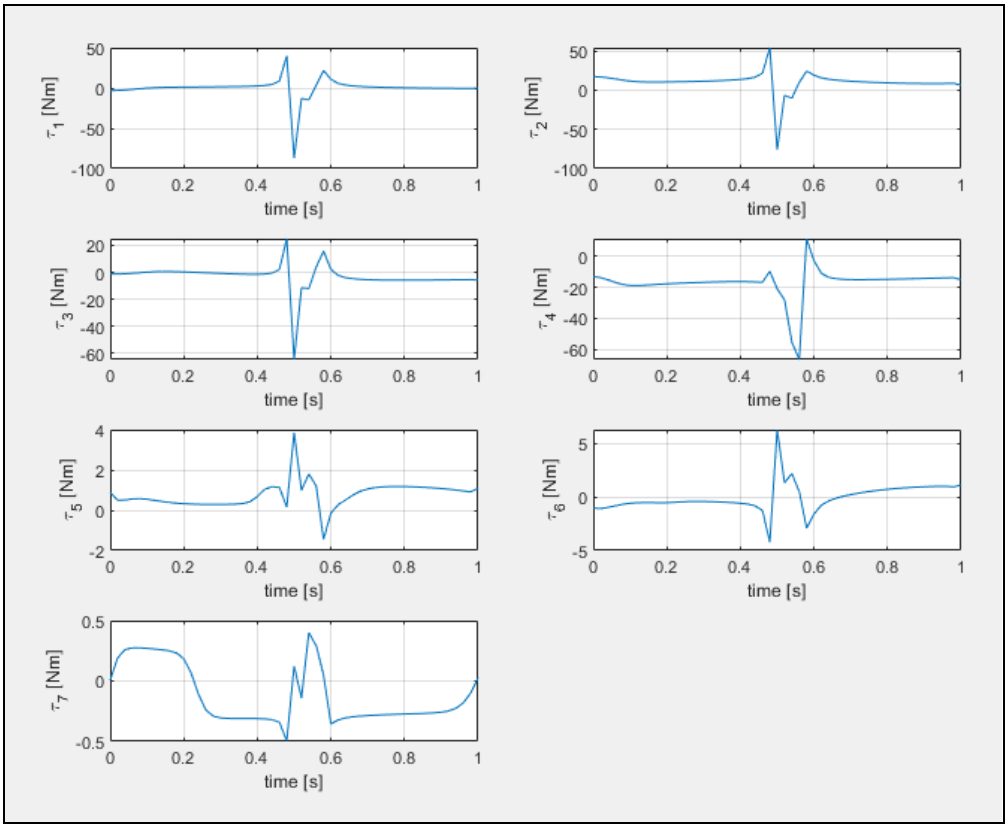

Figure 10: Joint Torques