Fundamentels of Robotics

Final Exam. Innopolis University, Fall 2020

Name: Hany Hamed
Group Number: BS18-Robotics

Contents

FoR, Innopolis University, Fall 2020
Author: Hany Hamed

Task1

The complete model is as following (in the notations of the lecture):

$T = [T_x\, T_y\, T_z\, R_x\, R_y\, R_z]_b\, R_x(q_1{+}\Delta q_1)[Ty\, T_z\, R_y\, R_z]_1\, T_x(q_2{+}\Delta q_2)\, [R_y\, R_z]_2\, T_z(q_3{+}\Delta q_3)\, [R_x\ R_y]_3\ [T_x\ T_y\ T_z\ R_x\ R_y\ R_z]_t$

Note: $[R_e]_i = R_{ei}$

We will remove:

(a) $R_x(\Delta q_1)$ as it will be added to the base

(b) R_{z2} as two consecutive orthogonal Prismatic joints

(c) $T_z(\Delta q_3)$ as it will be added to the tool

(d) R_{z2} as it can swapped with T_z of the prismatic joint as both of them on the same axis and then can be added to the tool

(e) $R_{x3},\ R_{y3}$ as they will be added to the tool

(f) $R_{xt},\ R_{yt},\ R_{zt}$ as it wil be removed as we are not measuring orientation information for the end effector but position for the references

Note: I mean by "added to" that it is swapped and can be identified with that parameter.

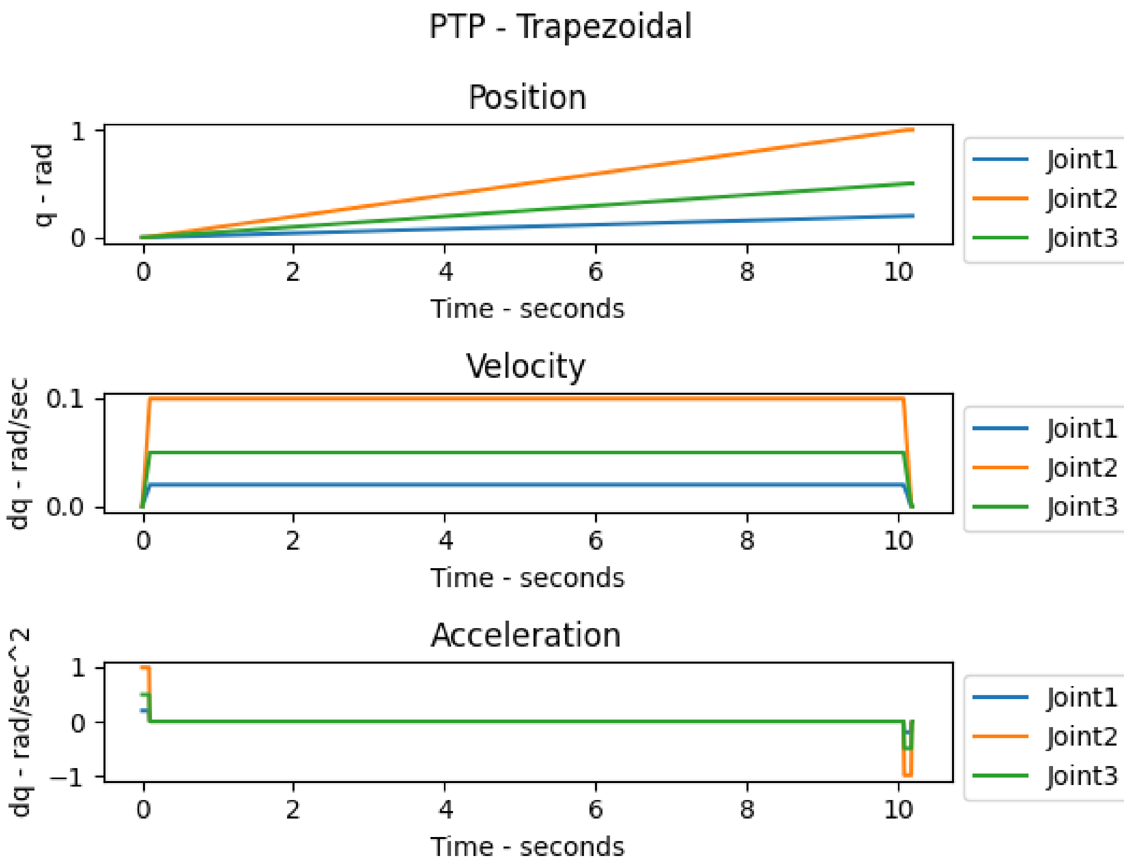
The irreducible model after eliminations:

$T = [T_x\, T_y\, T_z\, R_x\, R_y\, R_z]_b\, R_x(q_1)[Ty\, T_z\, R_y\, R_z]_1\, T_x(q_2{+}\Delta q_2)\, [R_y]_2\, T_z(q_3)[T_x\, T_y\, T_z]_t$

Then, we have six parameters for T_{robot}, six parameters for T_{base}, and 3 parameters for T_{tool}

Task2

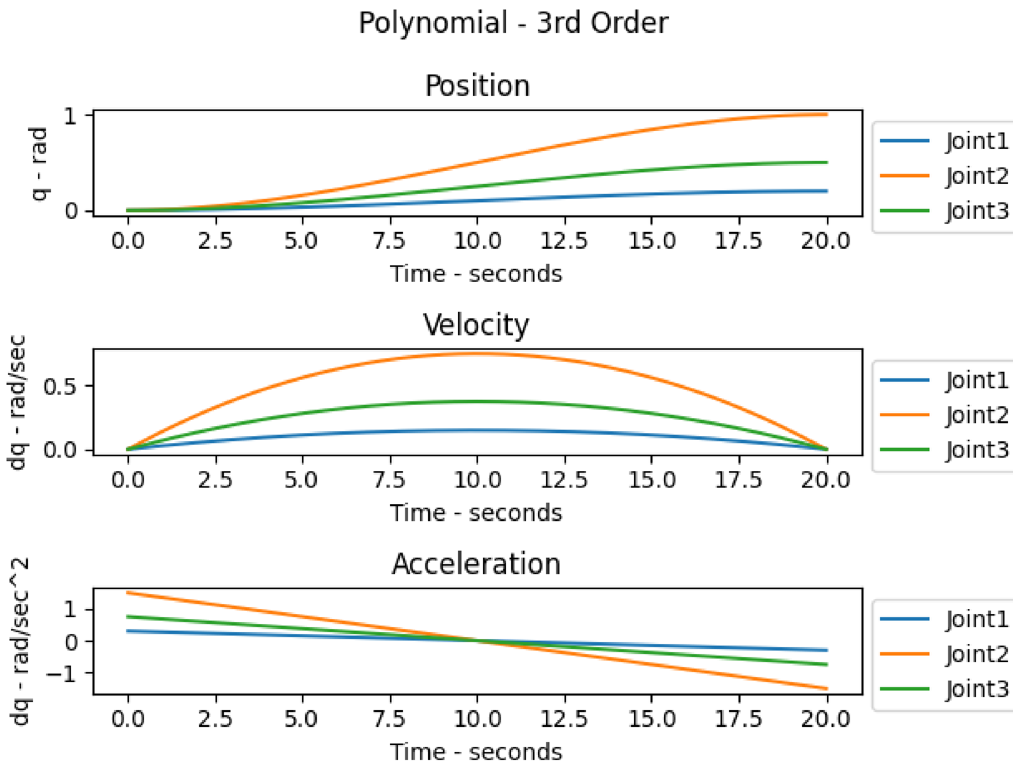
For PTP motion, I have used Trapezoidal profile, and the following figures determine the joint positions, velocities and accelerations.



PTP trajectories

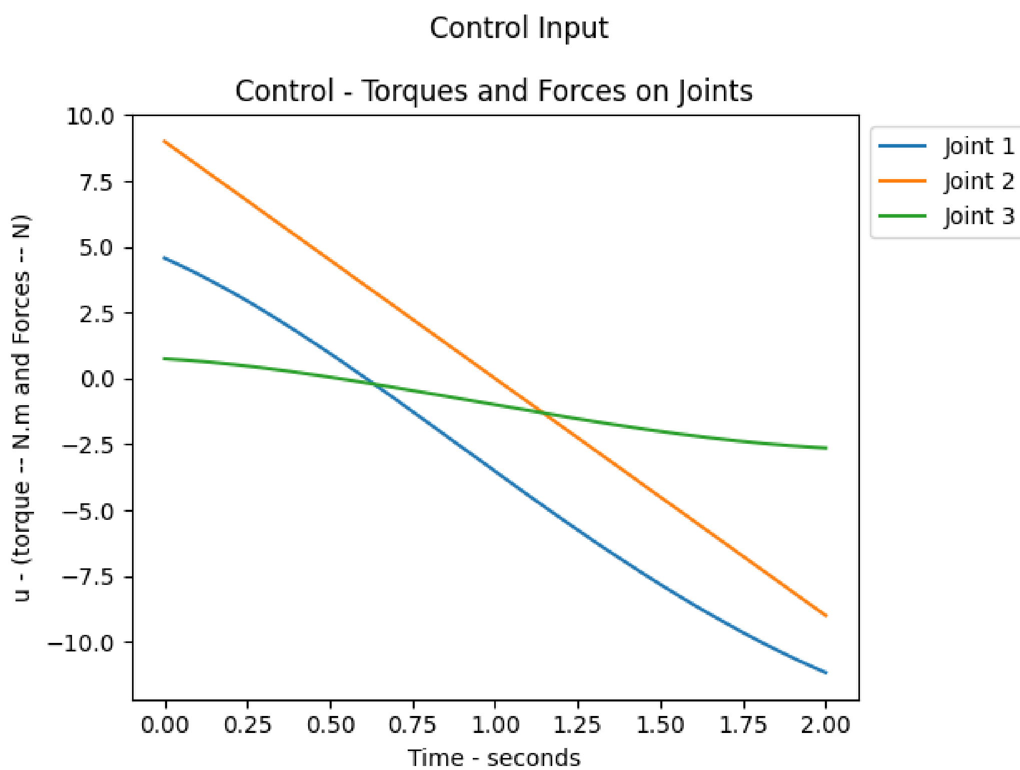
Task3

First, I have used 3rd order polynomial in order to satisfy the 4 constraints and generate a polynomial trajectory, and it is as the following:



Trajectory 3rd degree polynomial

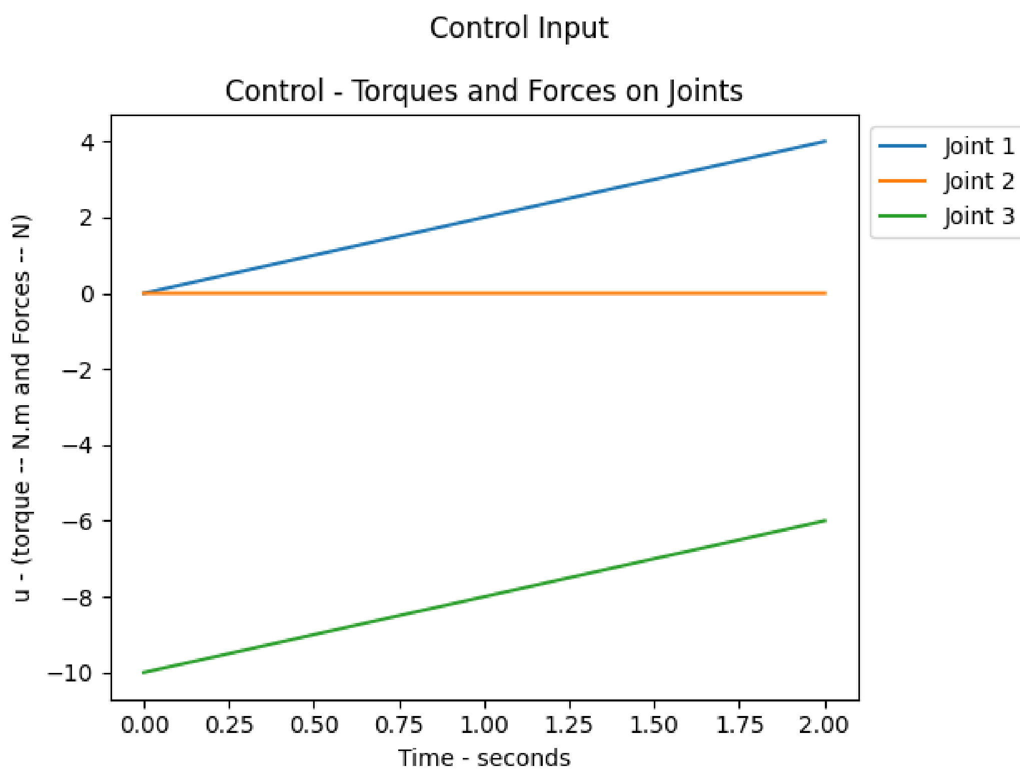
Then, using the inverse dynamics for Euler-Lagrange in order to get the torques and forces corresponds to this trajectory.



Torques and Forces plots

Task4

First, I have used 1st order polynomial in order to satisfy the 2 constraints on the force/torque equation and generate a polynomial trajectory (function depends on t), and it is as the following:



Force/Torque polynomial function with time

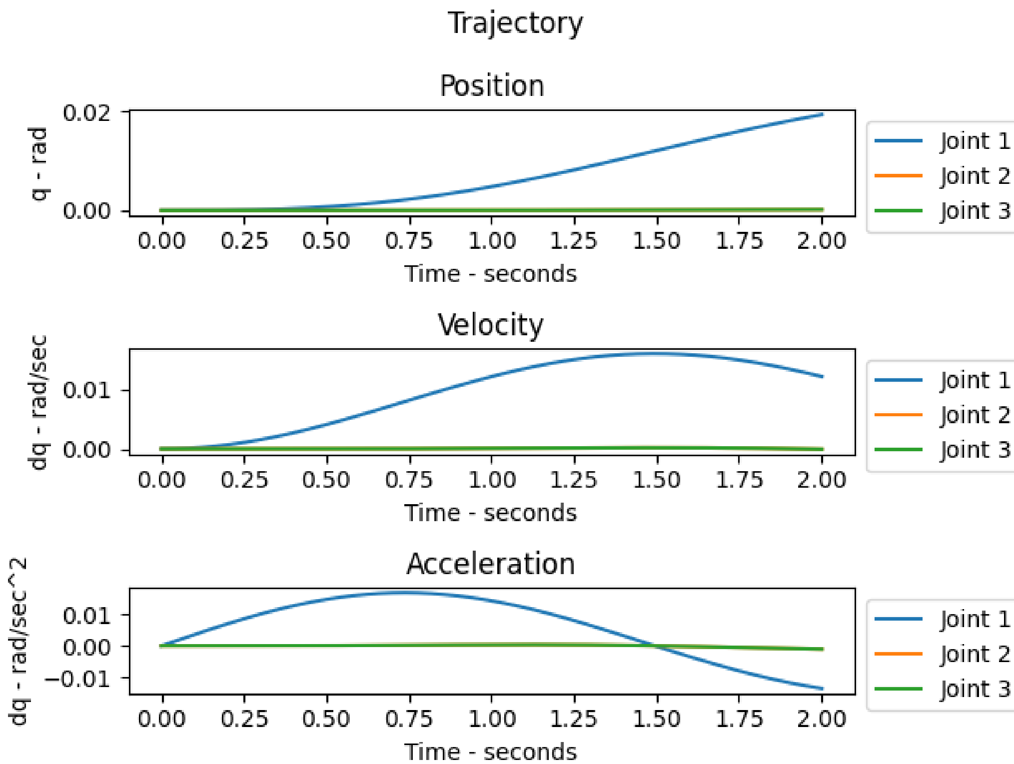
Then, using the direct dynamics for Euler-Lagrange in order to get the trajectory this function of time for forces and torques.

It was as following (Reference: here):

- numerical integration, at current state $(q, \dot{q})$, of
  $\ddot{q} = M^{-1}(q)[u - (c(q,\dot{q}) + g(q))] = M^{-1}(q)[u - n(q,\dot{q})]$
- Coriolis, centrifugal, and gravity terms
  $$n = NE_{{}^0 g}(q, \dot{q}, 0) \qquad \text{complexity } O(N)$$
- i-th column of the inertia matrix, for $i = 1, \ldots, N$
  $$M_i = NE_0(q, 0, e_i) \qquad O(N^2)$$
- numerical inversion of inertia matrix
  $$InvM = \text{inv}(M) \qquad O(N^3)$$
  but with small coefficient
- given u, integrate acceleration computed as
  $$\ddot{q} = InvM * [u - n] \quad \Rightarrow$$
  new state $(q, \dot{q})$
  and repeat over time ...

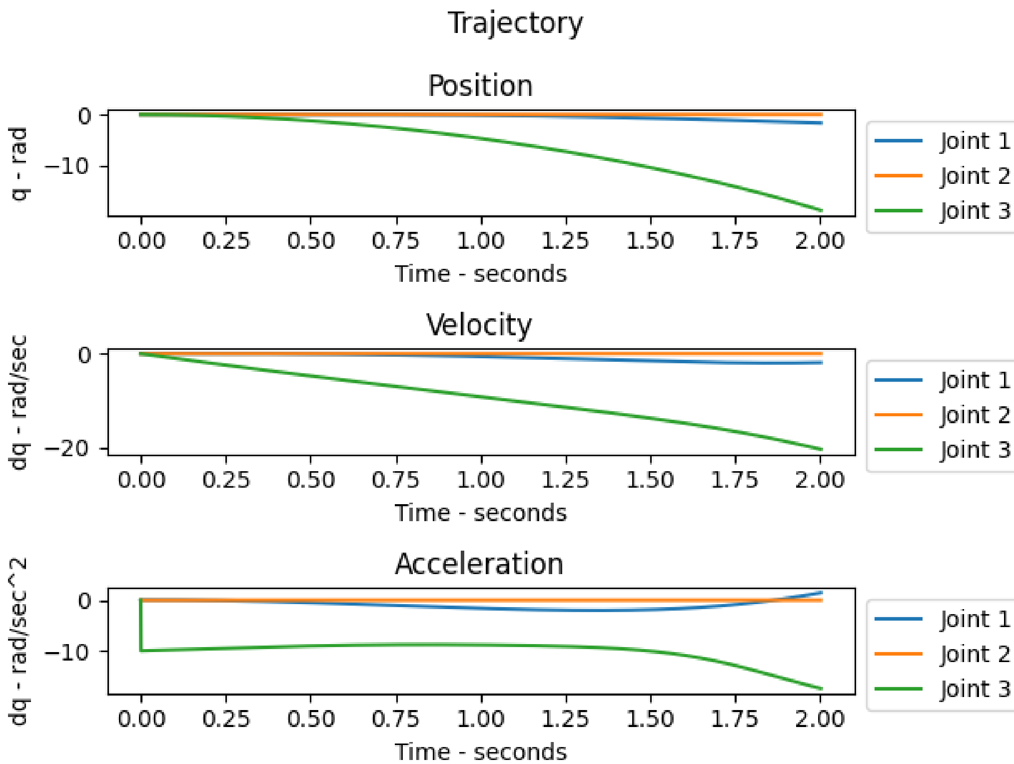
Method for Direct Dynamics based on the recursive solution of the inverse dynamics

The plots:



Torques and Forces plots

Moreover, I implemented the direct dynamics for Euler-Lagrange in order to get the trajectory this function of time for forces and torques.



Torques and Forces plots

Note: They should be the same or one of them had a problem but did not have time to find the error (most probably in Newton euler)