ADVANCED CONTROL SYSTEMS

Francesco Dal Santo
VR496940
Università degli studi di Verona
2023/2024

Contents

1 Assignment 1

1.1 Task

Compute DH table, direct kinematics, inverse kinematics, geometric Jacobian, analytical Jacobian by hand and cross-checking with Robotics toolbox.

1.2 Robot model

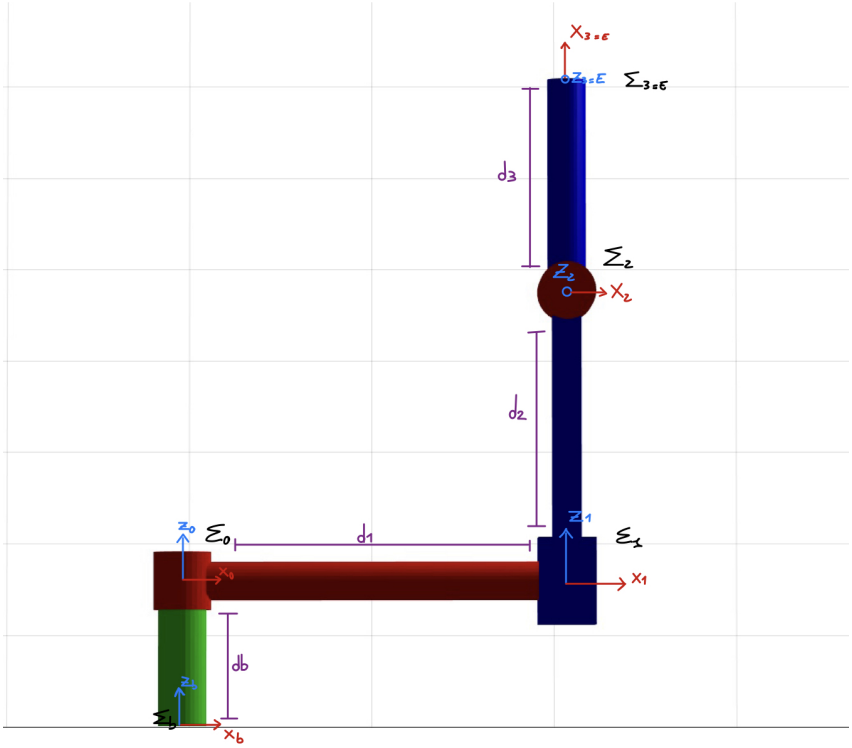


Figure 1: Robot model with frames

The d parameters representing the robot links lengths are:

- $d_0 = 0.15$
- $d_1 = 0.40$
- $d_2 = 0.30$
- $d_3 = 0.24$

1.3 DH Table

	θ	α	**d**	**a**
B-0	0	0	d_b	0
0-1	q_1	0	0	d_1
1-2	0	$\frac{\pi}{2}$	$d_2 + q_2$	0
2-E	$\frac{\pi}{2}+q_3$	0	0	d_3

Table 1: DH table of the robot model

1.4 Direct Kinematics

It'll follow a list of rotational matrices describing the changes of coordinates from the various frames Σ with respect to Σ_B:

$$T_0^B = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \mathrm{d_b} \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad T_1^B = \begin{pmatrix} \cos(q_1) & -\sin(q_1) & 0 & d_1 \cos(q_1) \\ \sin(q_1) & \cos(q_1) & 0 & d_1 \sin(q_1) \\ 0 & 0 & 1 & \mathrm{d_b} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$T_2^B = \begin{pmatrix} \cos(q_1) & 0 & \sin(q_1) & d_1 \cos(q_1) \\ \sin(q_1) & 0 & -\cos(q_1) & d_1 \sin(q_1) \\ 0 & 1 & 0 & d_2 + \mathrm{d_b} + q_2 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$T_E^B = \begin{pmatrix} \cos(q_1)\cos\left(q_3+\frac{\pi}{2}\right) & -\cos(q_1)\sin\left(q_3+\frac{\pi}{2}\right) & \sin(q_1) & d_1\cos(q_1) + d_3\cos(q_1)\cos\left(q_3+\frac{\pi}{2}\right) \\ \cos\left(q_3+\frac{\pi}{2}\right)\sin(q_1) & -\sin(q_1)\sin\left(q_3+\frac{\pi}{2}\right) & -\cos(q_1) & d_1\sin(q_1) + d_3\cos\left(q_3+\frac{\pi}{2}\right)\sin(q_1) \\ \sin\left(q_3+\frac{\pi}{2}\right) & \cos\left(q_3+\frac{\pi}{2}\right) & 0 & d_2 + \mathrm{d_b} + q_2 + d_3\sin\left(q_3+\frac{\pi}{2}\right) \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

It'll now follow a list of transformation from frame to frame:

$$T_0^B = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \mathrm{d_b} \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad T_1^0 = \begin{pmatrix} \cos(q_1) & -\sin(q_1) & 0 & d_1\cos(q_1) \\ \sin(q_1) & \cos(q_1) & 0 & d_1\sin(q_1) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$T_2^1 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & d_2 + q_2 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad T_E^2 = \begin{pmatrix} \cos\left(q_3 + \frac{\pi}{2}\right) & -\sin\left(q_3 + \frac{\pi}{2}\right) & 0 & d_3\cos\left(q_3 + \frac{\pi}{2}\right) \\ \sin\left(q_3 + \frac{\pi}{2}\right) & \cos\left(q_3 + \frac{\pi}{2}\right) & 0 & d_3\sin\left(q_3 + \frac{\pi}{2}\right) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

1.5 Inverse Kinematics

Given the direct kinematics after semplification:

$$p_E^B = \begin{pmatrix} \cos(q_1)\ (d_1 - d_3\ \sin(q_3)) \\ \sin(q_1)\ (d_1 - d_3\ \sin(q_3)) \\ d_2 + \mathrm{d_b} + q_2 + d_3\ \cos(q_3) \end{pmatrix} = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

We obtain the following inverse kinematics:

$$InvKin = \begin{pmatrix} \mathrm{atan2}(y, x) \\ -(d_2 + d_b + d_3 cos(q_3 sol)) + z \\ q_3 sol \end{pmatrix} = \begin{pmatrix} q_1 \\ q_2 \\ q_3 \end{pmatrix}$$

Where $q_3 sol$ is the the following:

$$q_3 sol = \begin{cases} \mathrm{asin}(-\frac{1}{d_3}(\frac{y}{\sin(q_1)} - d_1)) & if q_1 = \pi/2 \\ \mathrm{acos}(-\frac{1}{d_3}(\frac{y}{\cos(q_1)} - d_1)) & otherwise \end{cases}$$

The result was compared with the robotic toolbox one and was shown to be exactly the same.

1.6 Geometric Jacobian

Following:

$$J = \begin{bmatrix} z_0 \times (p_E^B \times p_0^B) & z_1 & z_2 \times (p_E^B \times p_2^B) \\ z_0 & 0 & z_2 \end{bmatrix}$$

With:

- $z_1 = R_1^B * z_0$
- $z_2 = R_2^b * z_1$

By cross-checking with the toolbox we see that the following result is correct, paying attention at the inverted linear and angular velocity shown in the toolbox.

$$J_G = \begin{pmatrix} -\sin(q_1)\ (d_1 - d_3\ \sin(q_3)) & 0 & -d_3\ \cos(q_1)\ \cos(q_3) \\ \cos(q_1)\ (d_1 - d_3\ \sin(q_3)) & 0 & -d_3\ \cos(q_3)\ \sin(q_1) \\ 0 & 1 & -d_3\ \sin(q_3) \\ 0 & 0 & \sin(q_1) \\ 0 & 0 & -\cos(q_1) \\ 1 & 0 & 0 \end{pmatrix}$$

Figure 6: Geometric Jacobian after simplification

1.7 Analytical Jacobian

We obtain the analytical Jacobian using the ZYZ euler angles T matrix:

$$J_A = T(\Phi)^{-1} J_G$$

with

$$R = \begin{pmatrix} 0 & -\sin(\phi) & \cos(\phi)\ \sin(\theta) \\ 0 & \cos(\phi) & \sin(\phi)\ \sin(\theta) \\ 1 & 0 & \cos(\theta) \end{pmatrix} \qquad T(\Phi) = \begin{pmatrix} I_3 & 0 \\ 0 & R \end{pmatrix}$$

obtaining :

$$J_A = \begin{pmatrix} -\sin(q_1)\ (d_1 - d_3\ \sin(q_3)) & 0 & -d_3\ \cos(q_1)\ \cos(q_3) \\ \cos(q_1)\ (d_1 - d_3\ \sin(q_3)) & 0 & -d_3\ \cos(q_3)\ \sin(q_1) \\ 0 & 1 & -d_3\ \sin(q_3) \\ 1 & 0 & \frac{\sin(\phi - q_1)\ \cos(\theta)}{\sin(\theta)} \\ 0 & 0 & -\cos(\phi - q_1) \\ 0 & 0 & -\frac{\sin(\phi - q_1)}{\sin(\theta)} \end{pmatrix}$$

Figure 8: Analytical Jacobian after simplification

2 Assignment 2

2.1 Task

Compute the Kinetic and Potential energy

2.2 Kinetic energy

I've assumed homogeneus density for every link, and calculated the mass based on:

$$\mathrm{m} = \mathrm{V} * \rho$$

obtaining the following masses:

$$V_1 = 3.5186[kg] \quad V_2 = 2.16[kg] \quad V_3 = 2.41[kg]$$

It'll follow the vectors which represent the position of the center of mass of each link with respect to it's frame:

$$p_{l1} = \begin{pmatrix} -\frac{d1}{2} \\ 0 \\ 0 \end{pmatrix} \quad p_{l12} = \begin{pmatrix} 0 \\ -\frac{d2}{2} \\ 0 \end{pmatrix} \quad p_{l2} = \begin{pmatrix} -\frac{d3}{2} \\ 0 \\ 0 \end{pmatrix}$$

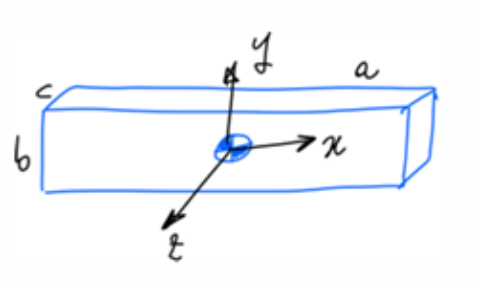
We then calculate then inertia matrices (w.r.t. CoM). In the case of a parallelepiped link we use:

$$I_C = \begin{pmatrix} \frac{1}{12}m(b^2+c^2) & 0 & 0 \\ 0 & \frac{1}{12}m(a^2+c^2) & 0 \\ 0 & 0 & \frac{1}{12}m(a^2+b^2) \end{pmatrix}$$

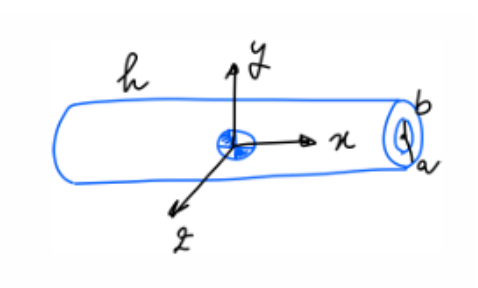
In the case of a cylindrical link we use:

$$I_C = \begin{pmatrix} \frac{1}{2}m(a^2+b^2) & 0 & 0 \\ 0 & \frac{1}{2}m(3(a^2+b^2)^2+h^2) & 0 \\ 0 & 0 & \frac{1}{2}m(3(a^2+b^2)^2+h^2) \end{pmatrix}$$

Following these:



(a) Parallelepiped



(b) Cylindrical

We then obtain the inertia I of every link as:

$$I = I_c + m(r^T r I_{3x3} - r r^T)$$

where

- m = mass of the link
- I_{3x3} = identity matrix
- r = distance of the CoM_i w.r.t frame Σ_i

We now compute the Partial Jacobians, knowing that

- p_{j-1} = position vector of the origin of Frame
- z_{j-1} = the unit vector axis z of Frame Σ_{j-1} w.r.t Σ_0

The columns of the Jacobian are:

$$J_{P_j}^{l_i} = \begin{cases} z_{j-1} & \text{prismatic joint} \\ z_{j-1} \times (p_{l_i} - p_{j-1}) & \text{revolute joint} \end{cases} \quad J_{O_j}^{l_i} = \begin{cases} 0 & \text{prismatic joint} \\ z_{j-1} & \text{revolute joint} \end{cases}$$

Finally we can compute the inertia matrix B(q):

$$\mathrm{B(q)} = \textstyle\sum_{i=1}^{n} (m_{li} {J_P^{l_i}}^T J_P^{l_i} + {J_O^{l_i}}^T R_i I_{l_i}^i R_i^T J_O^{l_i})$$

obtaining the following:

$$B(q) = \begin{pmatrix} 6.33\,{d_1}^2 - 2.41\,d_1\,d_3\,\sin(q_3) + 1.21\,{d_3}^2 \sin(q_3)^2 + 0.069 \sin(q_3)^2 + 0.232 & 0 & 0 \\ 0 & 4.57 & -1.21\,d_3\,\sin(q_3) \\ 0 & -1.21\,d_3\,\sin(q_3) & 1.21\,{d_3}^2 + 0.0695 \end{pmatrix}$$

And finally the total Kinetic Energy:

$$\mathrm{T}(q,\dot{q}) = \tfrac{1}{2} \dot{q}^T B(q) \dot{q}$$

the result can be found in the code given its excessive length.

2.3 Potential Energy

We can calculate the potential energy by adding all the single links contributions to each other, where every single contribution is expressed as:

$$U_{l_i} = -m_{l_i} g_0^T p_{l_i}$$

resulting in:

$$U = -34.3 * d_2 - 44.9 * q_2 - 11.8 * d_3 * cos(q_3)$$

3 Assignment 3

3.1 Task

Compute the equation of motion following the Lagrangian formulation.

3.2 B matrix

The computation for the B matrix can be found in the previous assignment.

$$B(q) = \begin{pmatrix} 6.33\,{d_1}^2 - 2.41\,d_1\,d_3\,\sin\left(q_3\right) + 1.21\,{d_3}^2\sin\left(q_3\right)^2 + 0.069\sin\left(q_3\right)^2 + 0.232 & 0 & 0 \\ 0 & 4.57 & -1.21\,d_3\,\sin\left(q_3\right) \\ 0 & -1.21\,d_3\,\sin\left(q_3\right) & 1.21\,{d_3}^2 + 0.0695 \end{pmatrix}$$

3.3 C matrix

We can find the C matrix by:

$$C_i(q,\dot{q}) = \sum_{j=1}^{n}\sum_{k=1}^{n}\frac{1}{2}(\frac{\delta b_{ij}}{\delta q_k} + \frac{\delta b_{ik}}{\delta q_j} + \frac{\delta b_{jk}}{\delta q_i})\dot{q_k}$$

by summing all the contributions we obtain the following matrix:

$$C(q,\dot{q}) = \begin{pmatrix} \mathrm{dq_3}\,\left(0.603\,\sin(2.0\,q_3)\,{d_3}^2 - 1.21\,d_1\,\cos(q_3)\,d_3 + 0.0345\,\sin(2.0\,q_3)\right) & 0 & \mathrm{dq_1}\,\left(0.603\,\sin(2.0\,q_3)\,{d_3}^2 - 1.21\,d_1\,\cos(q_3)\,d_3 + 0.0345\,\sin(2.0\,q_3)\right) \\ 0 & 0 & -1.21\,d_3\,\mathrm{dq_3}\,\cos(q_3) \\ -1.0\,\mathrm{dq_1}\,\left(0.603\,\sin(2.0\,q_3)\,{d_3}^2 - 1.21\,d_1\,\cos(q_3)\,d_3 + 0.0345\,\sin(2.0\,q_3)\right) & 0 & 0 \end{pmatrix}$$

3.4 G matrix

We can find the g matrix by:

$$g(q) = -\sum_{i=1}^{n} m_{l_i} g_0^T \frac{\delta p_{l_i}}{\delta q_i}$$

obtaining:

$$g(q) = \begin{pmatrix} 0 \\ 45.0 \\ -12.0\,d_3\,\sin\left(q_3\right) \end{pmatrix}$$

3.5 Equation of motion

We can finally find the equation of motion by:

$$\tau = B(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q)$$

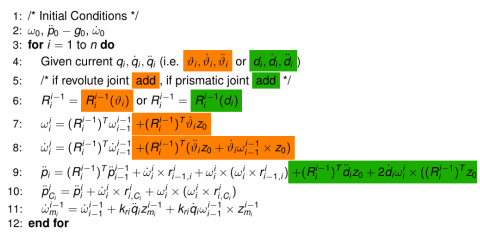
4 Assignment 4

4.1 Task

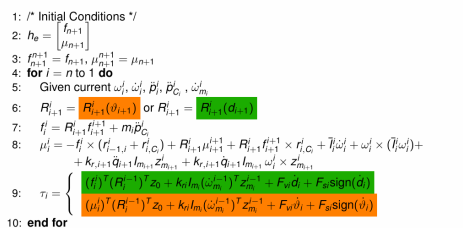
Compute the RNE formulation

4.2 Newton-Euler Formulation

We first execute the forward equations and then the backward equations:



(a) Algorithm 1: forward equations



(b) Algorithm 2: backward equations

We can then get the needed matrices as follows:

- $\tau_d = NE(q_d, \dot{q_d}, \ddot{q_d}, g_0)$
- $g(q) = NE(q, 0, 0, g_0)$
- $C(q, \dot{q}) = NE(q, \dot{q}, 0, 0)$
- $B_i(q) = NE(q, 0, e_i, 0)$

4.3 Results

It'll follow the comparison between lagrangian and RNE formulations given the following configuration:

- $q = (pi/6, -0.15, -pi/3)$
- $\dot{q} = (9, 14, -7)$
- $\ddot{q} = (-4, 2.5, 5)$

$$B_{lag} = \begin{pmatrix} 1.5 & 0 & 0 \\ 0 & 4.6 & 0.25 \\ 0 & 0.25 & 0.14 \end{pmatrix} \qquad B_{RNE} = \begin{pmatrix} 1.5 & 0 & 0 \\ 0 & 4.6 & 0.25 \\ 0 & 0.25 & 0.14 \end{pmatrix}$$

$$C_{lag}\dot{q} = \begin{pmatrix} 15.0 \\ -7.1 \\ 9.5 \end{pmatrix} \qquad C_{RNE}\dot{q} = \begin{pmatrix} 15.0 \\ -7.1 \\ 9.5 \end{pmatrix}$$

$$g_{lag} = \begin{pmatrix} 0 \\ 45.0 \\ 2.5 \end{pmatrix} \qquad g_{RNE} = \begin{pmatrix} 0 \\ 45.0 \\ 2.5 \end{pmatrix}$$

$$\tau_{lag} = \begin{pmatrix} 8.7 \\ 50.0 \\ 13.0 \end{pmatrix} \qquad \tau_{RNE} = \begin{pmatrix} 8.7 \\ 50.0 \\ 13.0 \end{pmatrix}$$

5 Assignment 5

5.1 Task

Compute the dynamic model in the operational space

5.2 Operational Space

The equations used for this assignment are the following:

- $B_A(x) = (J_a B^{-1} J_a^T)^{-1}$
- $C_A(x, \dot{x})\dot{x} = B_A J_A B^{-1} C\dot{q} - B_A \dot{J}_A \dot{q}$
- $g_A(x) = B_A J_A B^{-1} g$
- $u_e = T_A^T(x) h$

6 Assignment 6

6.1 Task

- Design the Joint Space PD control law with gravity compensation
- What happens if g(q) is not taken into account?
- What happens if the gravity term is set constant and equal to g(q_d) within the control law?
- What happens if q_d is not constant (e.g. $q_d(t) = \overline{q}_d + \Delta sin(\omega t)$)?

6.2 Schemes

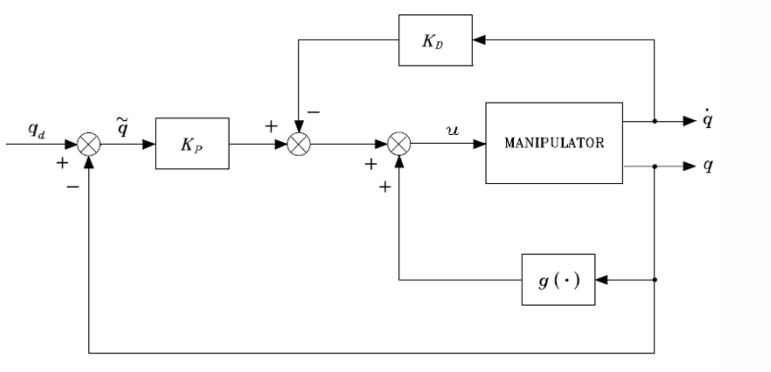


Figure 15: Joint Space PD control with gravity compensation block scheme

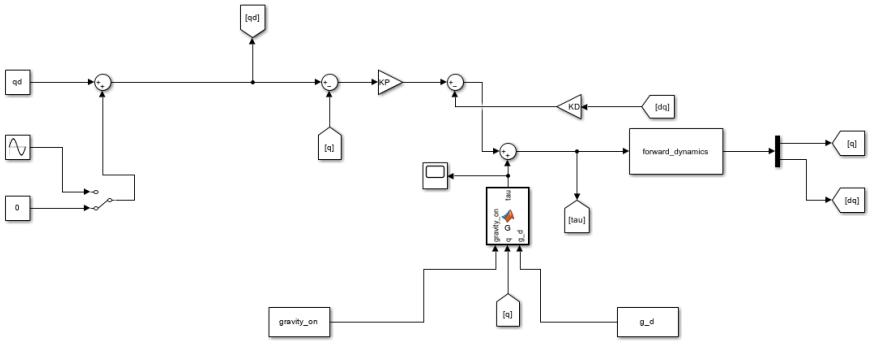


Figure 16: Simulink implementation of the Joint Space PD control with gravity compensation block scheme

6.3 Results

I chose as starting position [0;0;0] and as the desired final position $[\frac{\pi}{6};0.1;\frac{\pi}{3}]$

6.3.1 Gravity compensation on

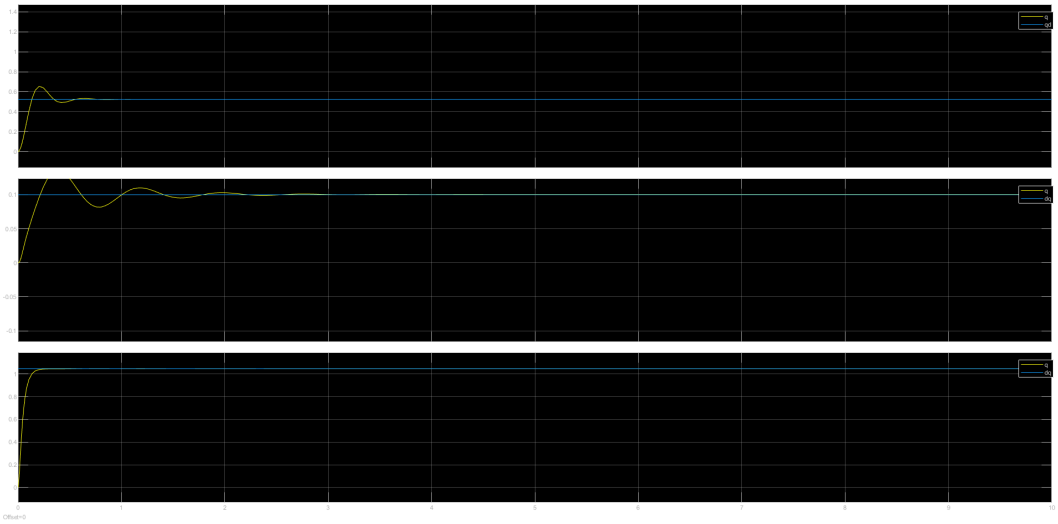


Figure 17: Results with gravity compensation active

As we can see we perfectly asymptotically reach the desired position in all the joints.

6.3.2 Graivity compensation off

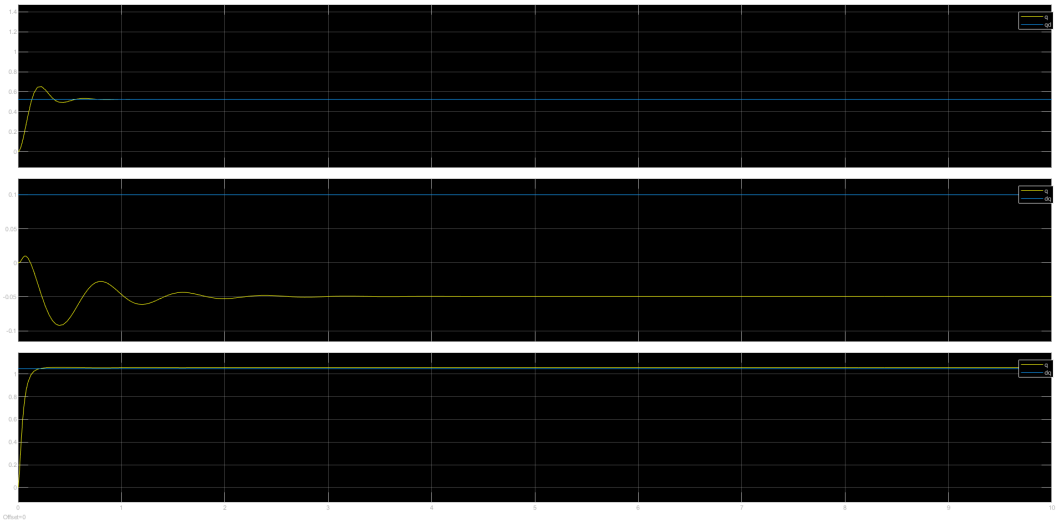


Figure 18: Results with gravity compensation not active

In this case we can see that the asymptotically result is not achieved for the second and third joint, which is what we expect since the gravity affects only those two and not the first one.

6.3.3 Constant desired gravity compensation

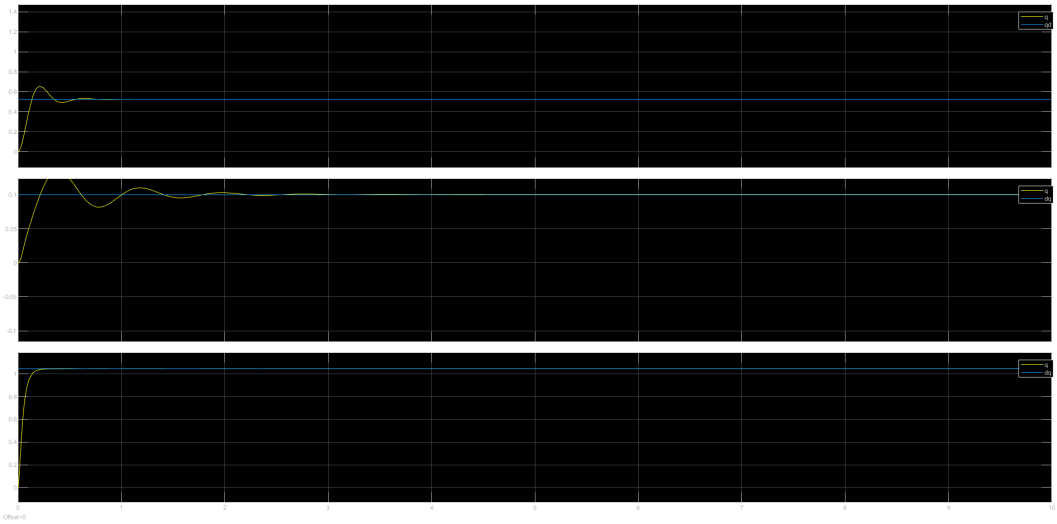


Figure 19: Results with constant gravity compensation

Here we applied a constant gravity compensation based on the final position and not on the current one, since the gravity term guarantees the final position it is expected, as we find in the plot, that at the end the desired positon is perfectly achieved

6.3.4 Not constant desired position

In the case of a not constant desired position we obtain the following result:

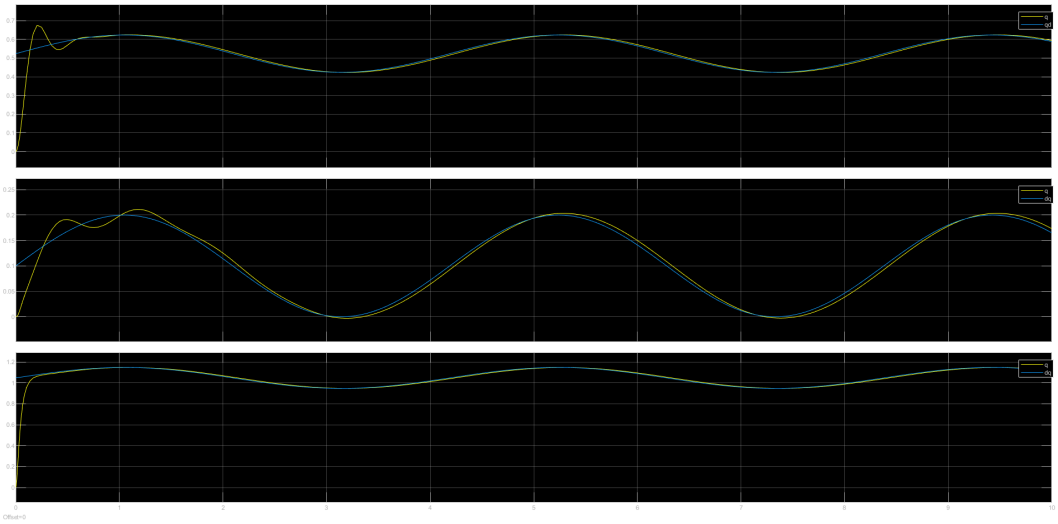


Figure 20: Results with gravity compensation active but not a constant desired position

7 Assignment 7

7.1 Task

- Design the Joint Space Inverse Dynamics Controls law
- Check that in the nominal case the dynamic behaviour behaviour is equivalent to the one of a set of stabilized double integrators
- Check the behaviour of the control law when the $\hat{B}, \hat{C}, \hat{g}$ used within the controller are different than the "true ones" B,C,g
- What happens to the torque values when the setting time of the equivalent second order systems is chosen very small

7.2 Schemes

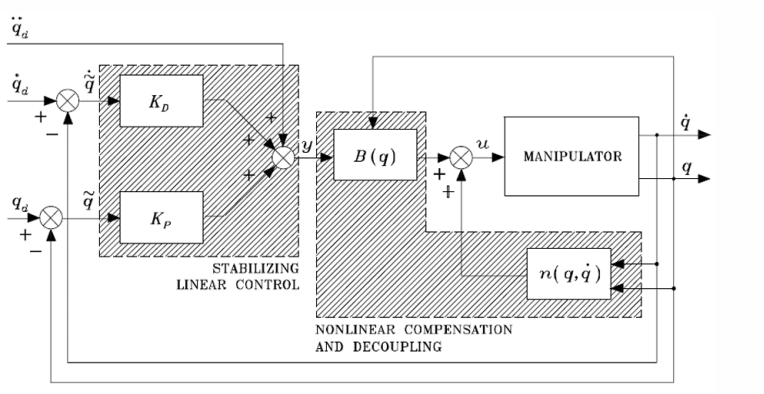


Figure 21: Joint Space Inverse Dynamics Control law block scheme

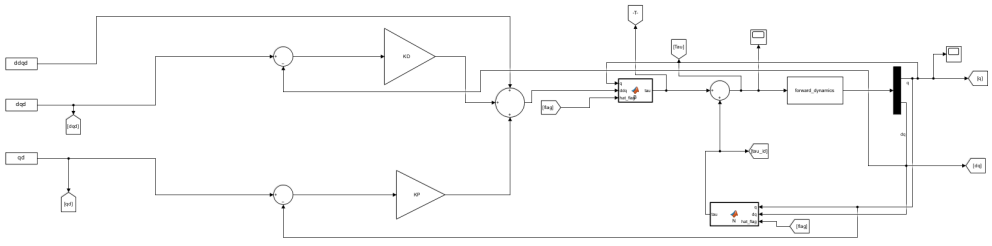


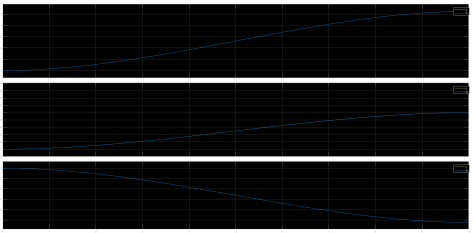
Figure 22: Simulink implementation of the Joint Space Inverse Dynamics Control law block scheme

7.3 Results

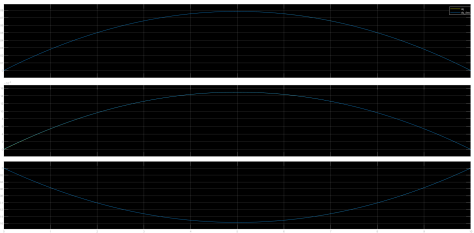
I chose as starting position [0;0;0] and as the desired final position $[\frac{\pi}{6};0.1;-\frac{\pi}{3}]$

7.3.1 Correct model

Here we can see that by using the correct matrices B,C,g we can obtain a perfect match between the desired and actual position.



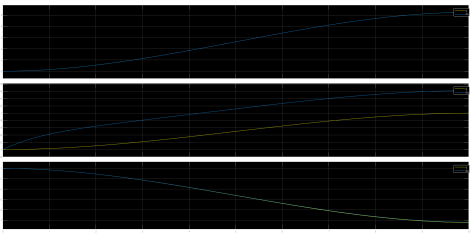
(a) positions plot for correct matrices



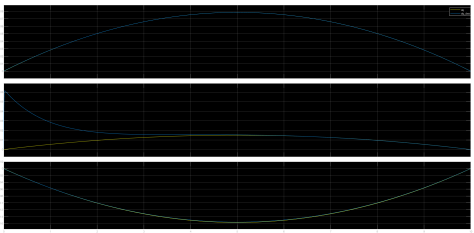
(b) velocities plot for correct matrices

7.3.2 Wrong model

Here we can see that by using the altered matrices $\hat{B}, \hat{C}, \hat{g}$ we lose the overlap, and as expected the second and third joint don't respect the final desired position as the g matrix which is responsible for that is not the correct one.



(a) positions plot for altered matrices



(b) velocities plot for altered matrices

7.4 Small execution time

If we set our desired time to reach the final position small, in this case is one second, we can observe some instantaneous peaks in the torques given to the joints (especially the second one in this case).

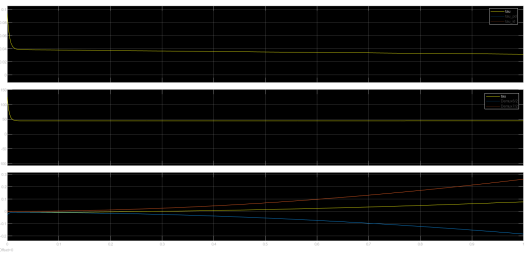


Figure 25: Torques for t=1

8 Assignment 8

8.1 Task

Implement in Simulink the Adaptive Control law for a 1-DoF link under gravity. Choose the dynamic parameters and their initial estimates, and set the desired trajectory as

- $q_d(t) = Asin(\omega t)$
- $\ddot{q_d}(t) =$ periodic square wave $\pm A$

8.2 Schemes

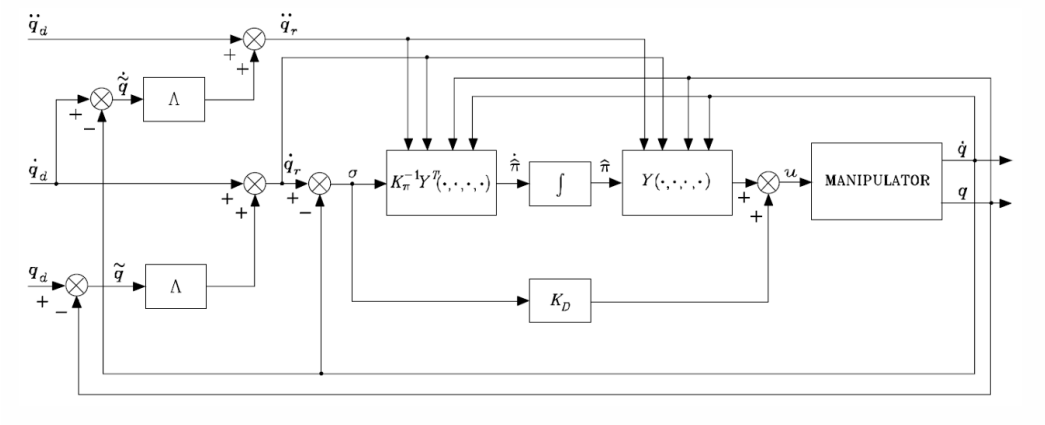


Figure 26: Joint Space Adaptive Control law block scheme

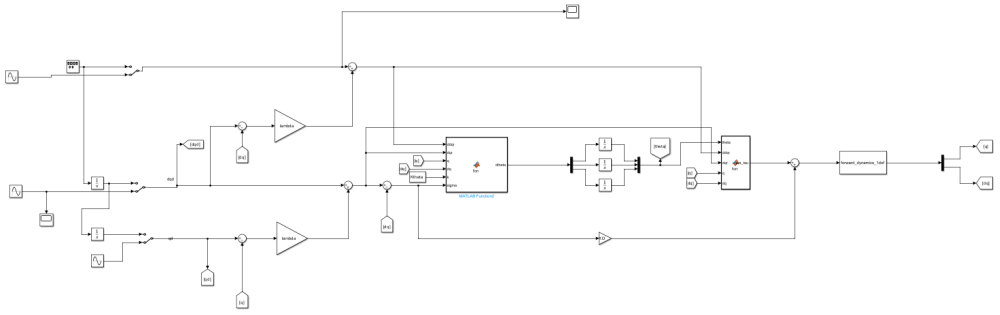
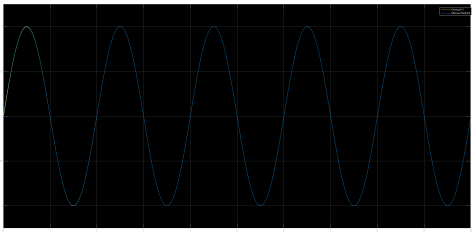


Figure 27: Simulink implementation of Joint Space Adaptive Control law block scheme

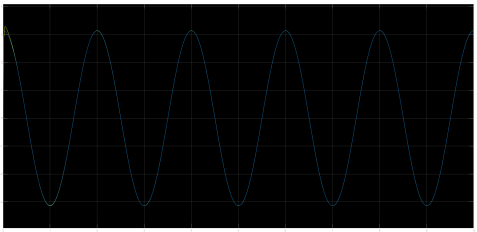
8.3 Results

8.3.1 Sinusoidal

For the sinusoidal position we obtain a perfect match in the desired position:



(a) positions plot sinusoidal input



(b) velocities plot for sinusoidal input

with the following estimations:

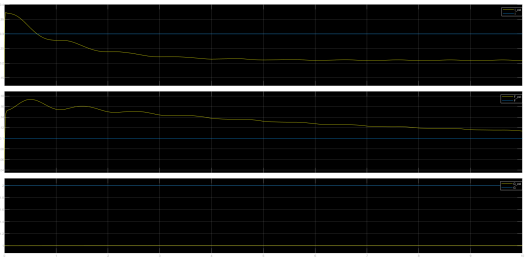
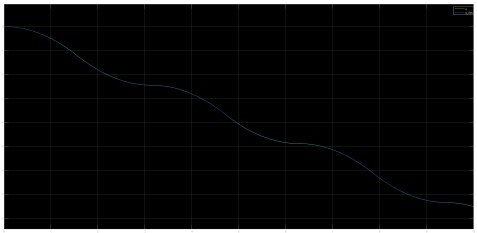


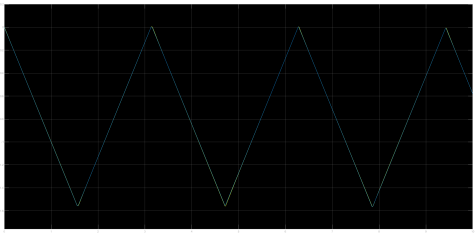
Figure 29: I,F,G and the estimated ones

8.3.2 Square wave

For the square wave acceleration we obtain a perfect match in the desired position:



(a) positions plot square wave input



(b) velocities plot for square wave input

with the following estimations:

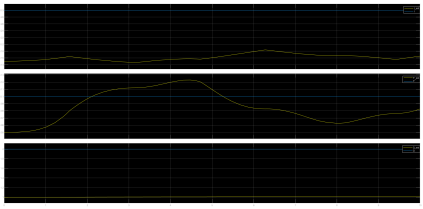


Figure 31: I,F,G and the estimated ones

9 Assignment 9

9.1 Task

Design the Operational Space PD control law with gravity compensation

9.2 Schemes

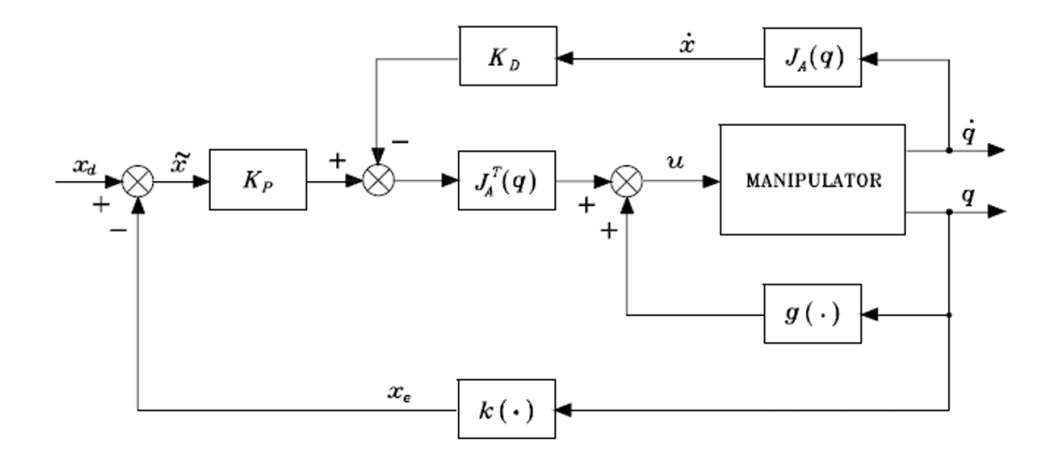


Figure 32: Operational Space PD control with gravity compensation block scheme

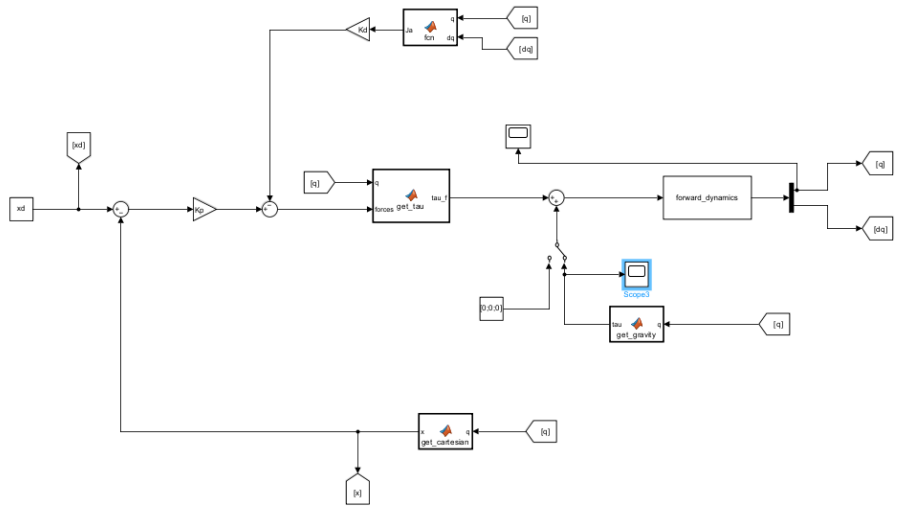


Figure 33: Simulink implementation of Operational Space PD control with gravity compensation block scheme

9.3 Results

It was chosen as starting joint configuration [0;0;0] and as final $[\frac{pi}{3};0.4;-\frac{pi}{2}]$ then converted into cartesian space.

9.3.1 Gravity on

Here we can see that the desired position in the cartesian space is perfectly reached in all x,y and z.

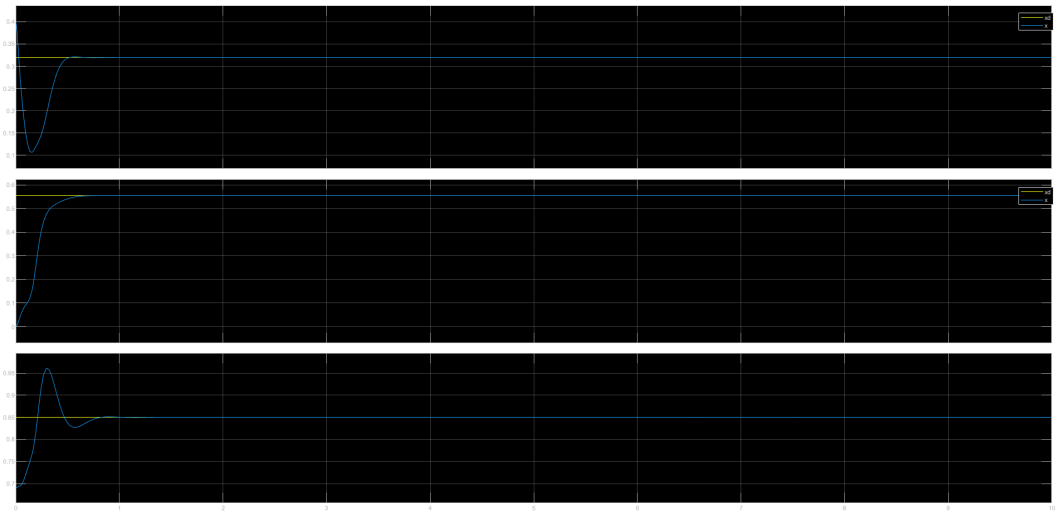


Figure 34: Position plot with gravity compensation active

9.3.2 Gravity off

Here we can see that the desired position in the cartesian space is not reached,especially in z, but this is expected given that the second joint which affects the z position is not applying the required torque to compensate the gravity effect.

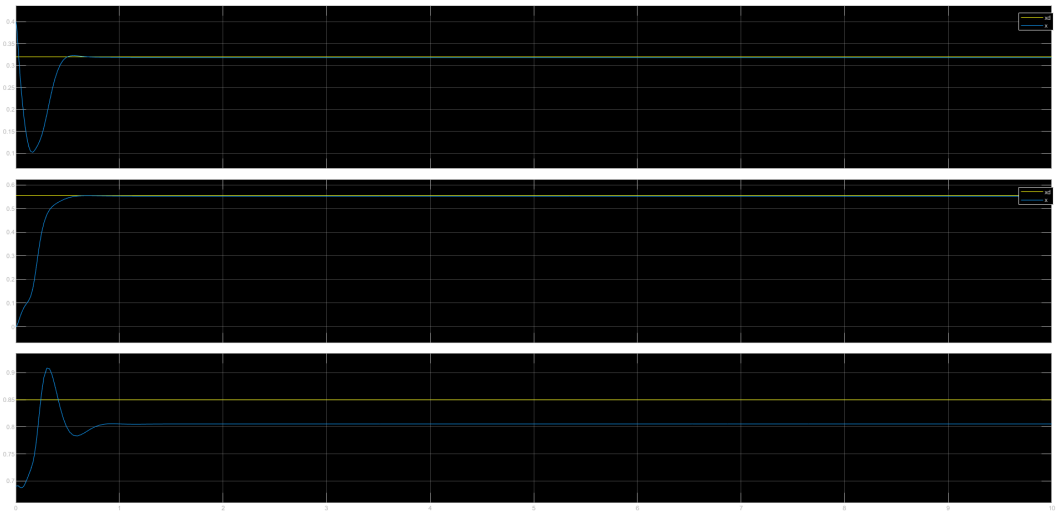


Figure 35: Position plot with gravity compensation turned off

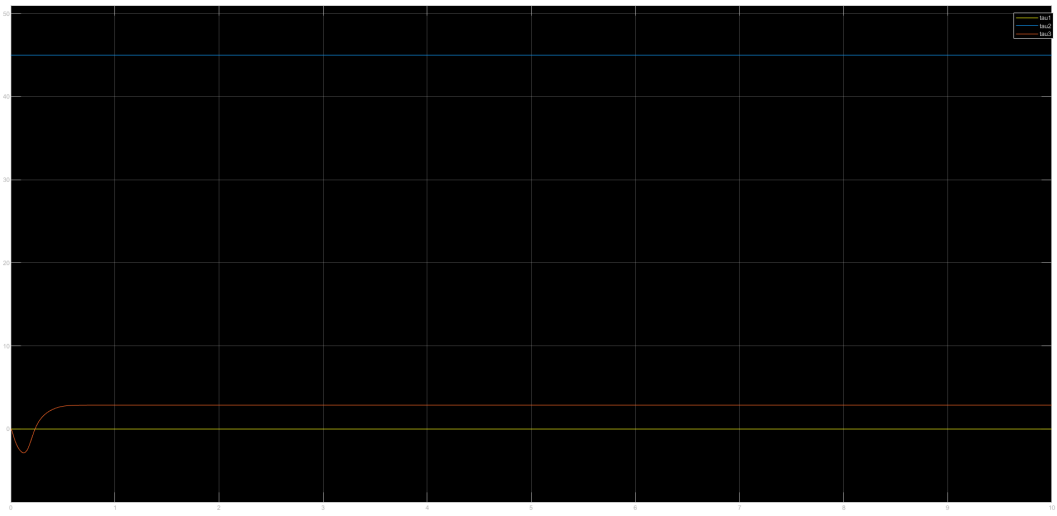


Figure 36: Tau required for all joints

10 Assignment 10

10.1 Task

Design the Operational Space Inverse Dynamics Control law

10.2 Schemes

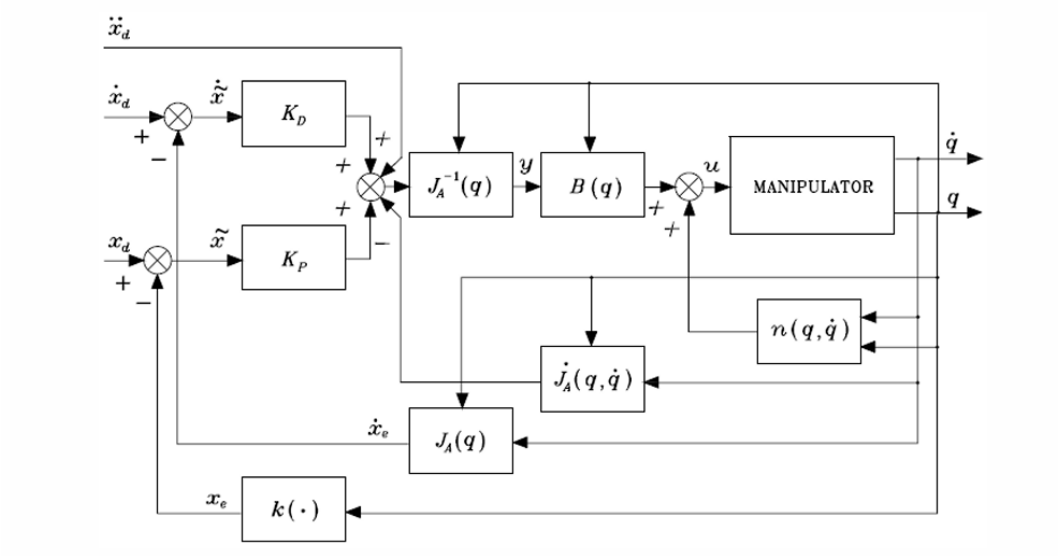


Figure 37: Operational Space Inverse Dynamics Control law block scheme

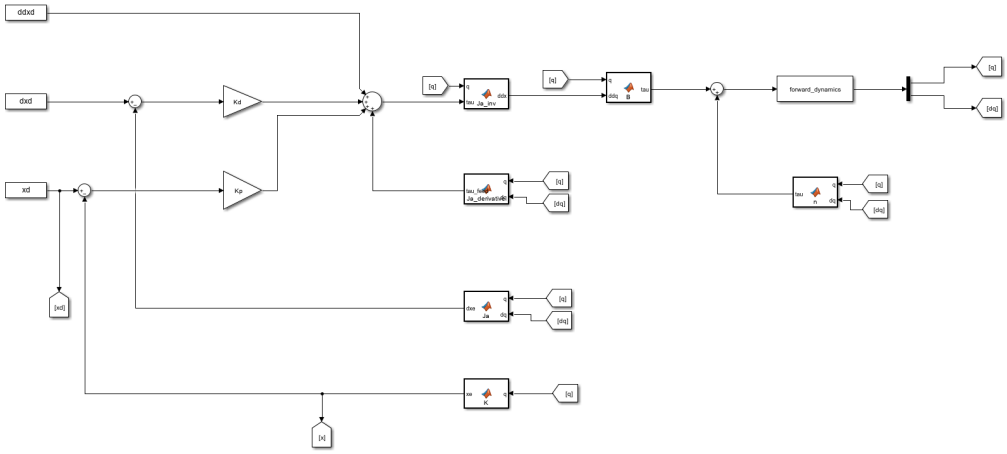


Figure 38: Simulink implementation of Operational Space Inverse Dynamics Control law block scheme

10.3 Results

A cubical trajectory was calculated in the joint space from [0;0;0] to $[\frac{-\pi}{3};0.3;\frac{\pi}{2}]$ and then converted into the cartesian space. As we can see in the following images we obtain as expected a perfect tracking of the desired trajectery in the position and orientation of the end effector

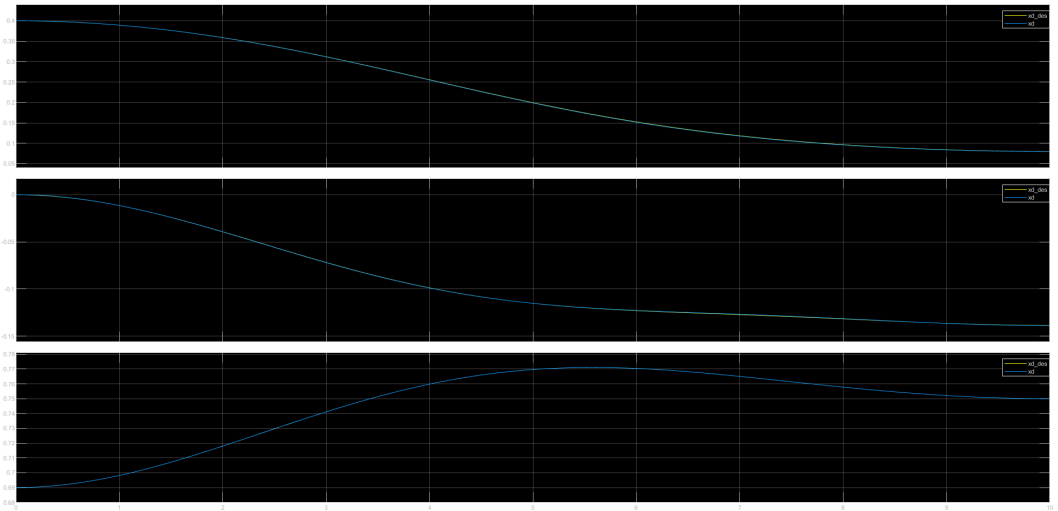


Figure 39: EE position

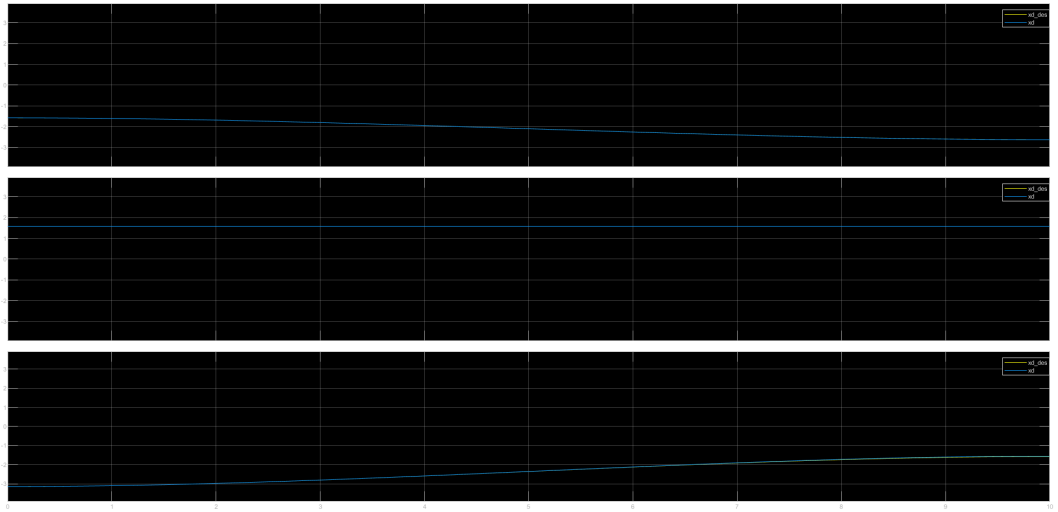


Figure 40: EE orientation

11 Assignment 11

11.1 Task

Study the compliance control. Simulate the two "extreme" cases when the end-effector is interacting with the environment(considered as a planar surface)

11.2 Theory

We report the control law for the compliance control:

$$\tau = g(q) + J_{A_d}^T(q, \tilde{x})(K_P\tilde{x} - K_D(q, \tilde{x})\dot{q})$$
$$\tilde{x} = x_d - x_e = -\begin{bmatrix} o_{d,e}^d \\ \phi_{d,e} \end{bmatrix}$$
$$J_{A_d} = T_a^{-1}(\phi_{d,e}) \begin{bmatrix} R_d^T & 0 \\ 0 & R_d^T \end{bmatrix}$$

The planar environment was designed along the z direction considering the configuration of my robot. The external wrench was computed on z by calculating the displacement between environmental rest position and the EE position:

$$h_e = Kx_{r,e}$$

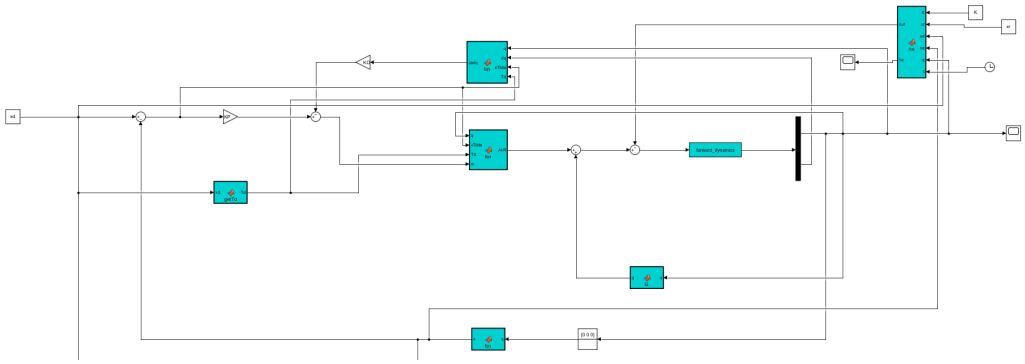


Figure 41: Simulink implementation of Compliance control block scheme

11.3 Results

11.3.1 Case K>>KP

In this case the environment stops completely the end effector, in this case we set $K_P = 50$ and K = 500.

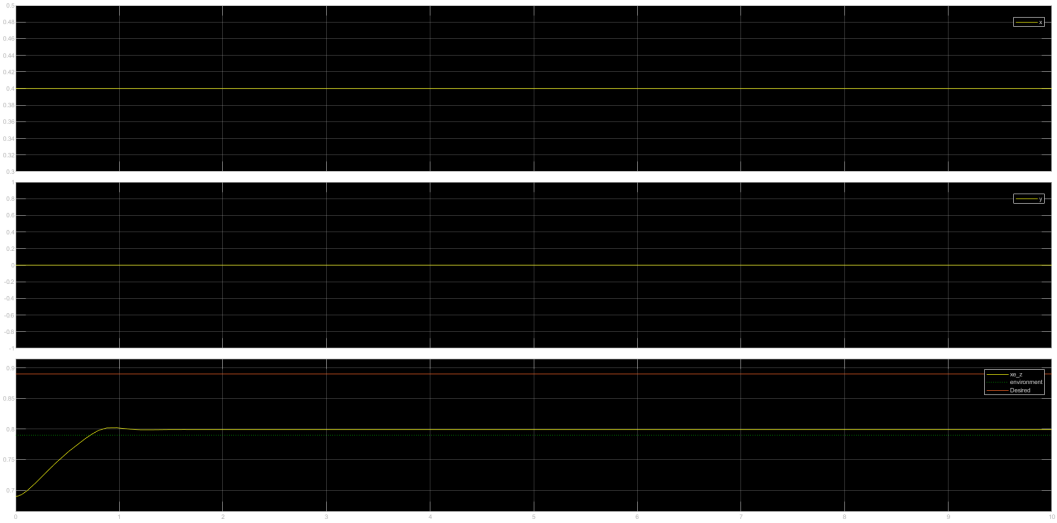


Figure 42: Environment >> Robot

11.3.2 Case K=KP

In this case the environment can't completely stop the robot, and since K is equal to K_p (50) the EE stops at halfway between the desired position and the rest position of the environment

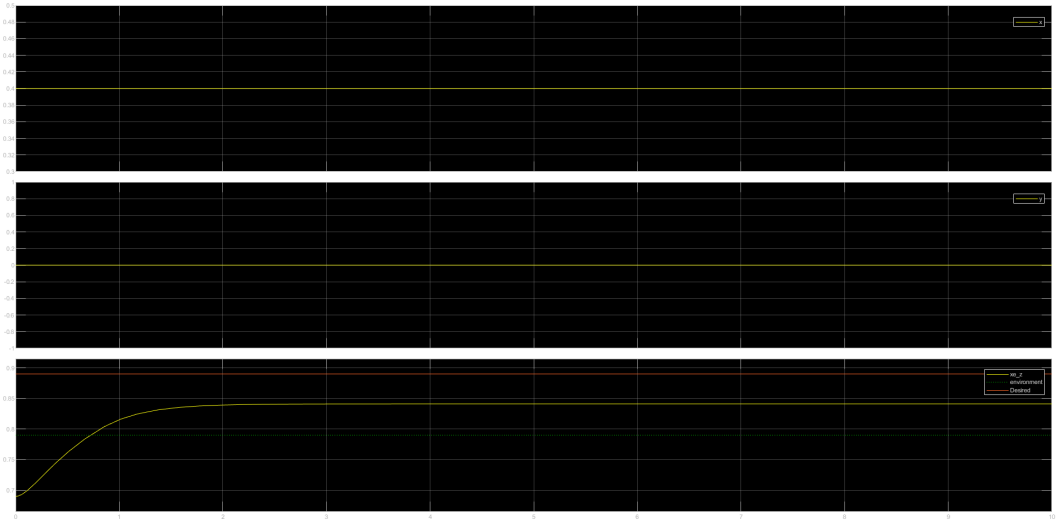


Figure 43: Environment = Robot

11.3.3 Case K<<KP

In this we set K = 1 and $K_p = 50$. We can see that the environment can't stop the robot as it reaches its desired position

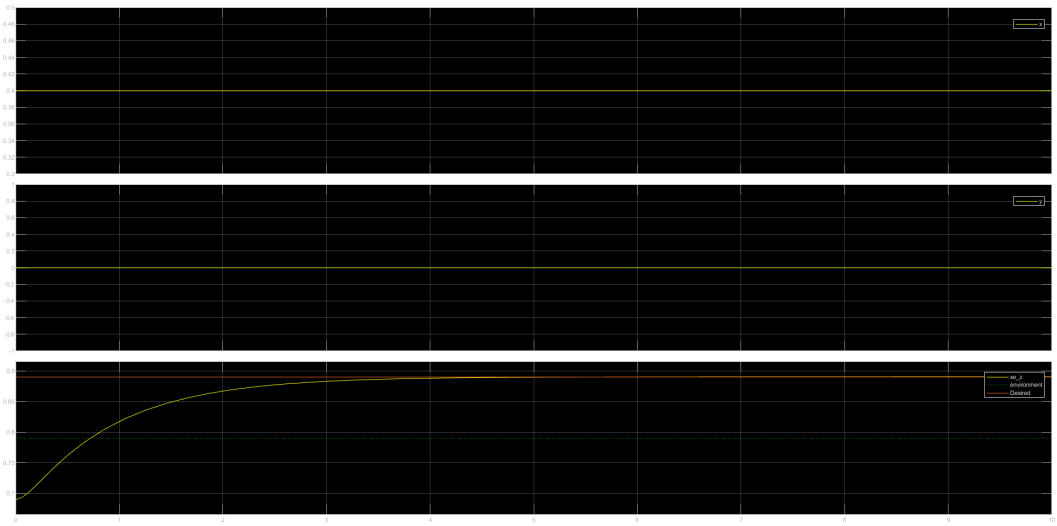


Figure 44: Environment << Robot

References