Manipulator Robots

Integrated Master Degree in Mechanical Engineering

Scientific Area of Control, Automation, and Industrial Informatics

UR3 Dynamics and Control

Author:
Luís Miranda, 81089
Diogo Catarino, 81772

Instructor:
Prof. Jorge Martins

June 3, 2018

Contents

1 Introduction

In this second part, firstly, the dynamics parameters will be calculated with the help of a *Solidworks* model and a dynamic model will be presented, implemented and validated. After this, a Decentralized PID Controller and a Centralized Inverse Dynamics Controller will be designed. Finally, a trajectory will be performed and the performance/results of both controllers will be compared and analysed.

2 Dynamics Parameters

First, it was necessary to build a *Solidworks* model of the manipulator. This model was developed with the help of the UR3 manual [2] and its corresponding dimensions. The figure 1 shows the model used for the calculus of the dynamics parameters.

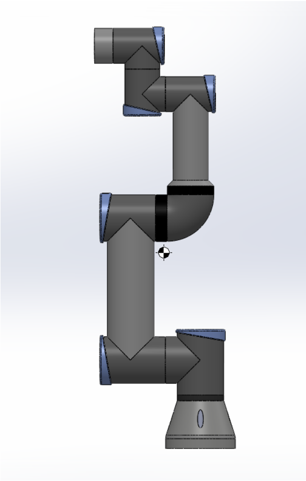

Figure 1: Solidworks model of the manipulator

With the help of the *MassProperties* tool in the *Evaluate* tab, it was possible to get all the manipulator links and motor parameters.

Link_i	Mass[Kg]	Center of Mass in Reference Frame_i [m]
1	0.57	[0 ; -0.010 ; 0.013]
2	1.36	[-0.128 ; 0 ; 0.087]
3	1.05	[-0.118 ; 0 ; 0.010]
4	0.32	[0 ; 0.006 ; 0.010]
5	0.34	[0 ; -0.006 ; 0.013]
6	0.16	[0 ; 0 ; -0.016]
Motor	0.10	[0 ; 0 ; 0,075]

Table 1: Manipulator Links and Motor Parameters

The remaining parameters needed for the dynamic model are the Inertia tensors, which can be seen in the appendix A.

3 Dynamics Model

There are 2 methods which gives us the dynamic model of the manipulator: the Lagrange formulation and the Newton−Euler formulation. The Lagrange formulation uses the total Lagrangian of the system, while the Newton−Euler formulation is based on a balance of all the forces acting on the generic link of the manipulator. We decided to use the Newton−Euler formulation because it is easier to implement computationally.

3.1 Newton-Euler Formulation

Using a set of equations, (7.107) to (7.114) from the book [3], allows a recursive type of solution; a forward recursion is performed for propagating link velocities and accelerations, followed by a backward recursion for propagating forces.

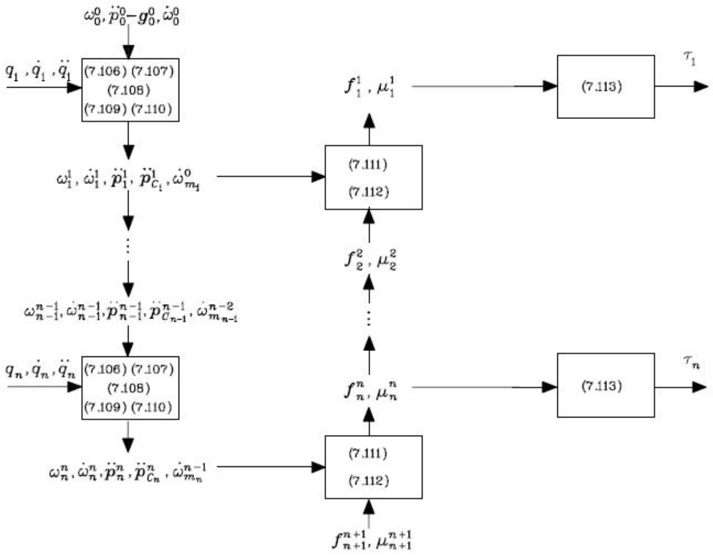
$\omega_0^0, \ddot{p}_0^0 - g_0^0, \dot{\omega}_0^0$
$q_1, \dot{q}_1, \ddot{q}_1$
(7.106) (7.107)
(7.108)
(7.109) (7.110)
$\omega_1^1, \dot{\omega}_1^1, \ddot{p}_1^1, \ddot{p}_{C_1}^1, \dot{\omega}_{m_1}^0$
f_1^1, μ_1^1
(7.113)
τ_1
(7.111)
(7.112)
f_2^2, μ_2^2
$\omega_{n-1}^{n-1}, \dot{\omega}_{n-1}^{n-1}, \ddot{p}_{n-1}^{n-1}, \ddot{p}_{C_{n-1}}^{n-1}, \dot{\omega}_{m_{n-1}}^{n-2}$
$q_n, \dot{q}_n, \ddot{q}_n$
(7.106) (7.107)
(7.108)
(7.109) (7.110)
f_n^n, μ_n^n
(7.113)
τ_n
$\omega_n^n, \dot{\omega}_n^n, \ddot{p}_n^n, \ddot{p}_{C_n}^n, \dot{\omega}_{m_n}^{n-1}$
(7.111)
(7.112)
$f_{n+1}^{n+1}, \mu_{n+1}^{n+1}$

Figure 2: Computational structure of the Newton-Euler recursive algorithm

The outputs of the algorithm are the binaries in each joint τ_i, which can be arranged in the form of the equation 1. In this equation, $B(q)$ is the inertia matrix, $C(q, \dot{q})$ is the quadratic velocity matrix and $g(q)$ is the gravitational contribution matrix.

$$\tau = B(q) \cdot \ddot{q} + C(q, \dot{q}) \cdot \dot{q} + F_v \cdot \dot{q} + F_s \cdot sgn(\dot{q}) + g(q) + J^T(q) \cdot h_e \qquad (1)$$

Simplifying the contribution of joint Coulomb and viscous friction, as well as of the forces and moments applied on the environment we ended up with the equation 2.

$$\tau = B(q) \cdot \ddot{q} + C(q, \dot{q}) \cdot \dot{q} + g(q) \qquad (2)$$

3.2 Validation of the model

The model was implemented on Simulink with the equation 2 as guideline, which gives us the components that characterize the dynamic model of the manipulator. Now the model needed to be validated by confronting it with the reality. For this validation, we used two test: one with all the torques null and all the joint angles null; and another with all the torques null and the second joint with an initial angle of $\vartheta_2 = \pi/2$.

The figure 3 shows the behavior of the manipulator. As can be see, with no torques applied, the manipulator is in rest with all joint angles null.

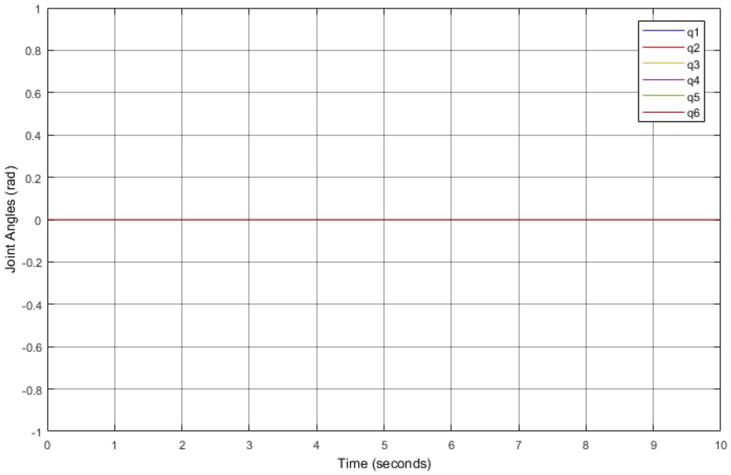

Figure 3: Joint angles values with no torque applied and initial null angles

The second test corresponded in letting the manipulator free from a initial position, $\vartheta_2 = \pi/2$, and observe its behavior as it falls. As expected, the manipulator presented a behavior like a gravitational pendulum, figure 4.

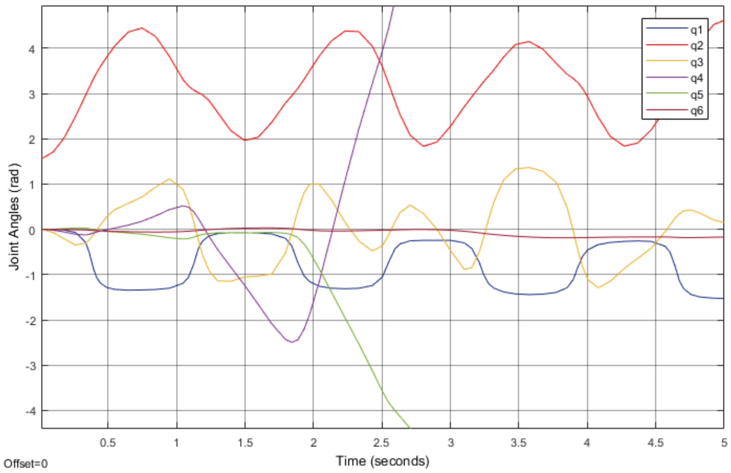

Figure 4: Joint angles values for the gravitational pendulum test

Since the system as no dissipation of energy, the sinusoidal behavior is visible. Also, because we have no dissipation of energy, some of the angles start increasing its value due to a whip effect.

4 Decentralized PID Control

4.1 Worst-case inertia configuration

The worst−case configuration for each joint has more than one possible solution. The joints which influence this solution do it with sine and cosine coefficients. Due to this, only one possible solution will be presented in the table 2.

Link$_i$	Worst Configuration[]	B$_{i\ max}$[m]
1	$[0 \quad \pi/2 \quad 0 \quad 0 \quad 0 \quad 0]$	0.4246
2	$[0 \quad 0 \quad 0 \quad 0 \quad \pi/2 \quad 0]$	0.3988
3	$[0 \quad 0 \quad 0 \quad 0 \quad \pi/2 \quad 0]$	0.09708
4	$[0 \quad 0 \quad 0 \quad 0 \quad \pi/2 \quad 0]$	0.02295
5	$[0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0]$	0.01804
6	$[0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0]$	0.01698

Table 2: Worst case Configuration

4.2 Design of the PID type controller

The theory behind designing a decentralized controller is that each joint is independent from each other. The influence of one joint to the others is considered to be disturbance in its dynamic model, which you can see bellow.

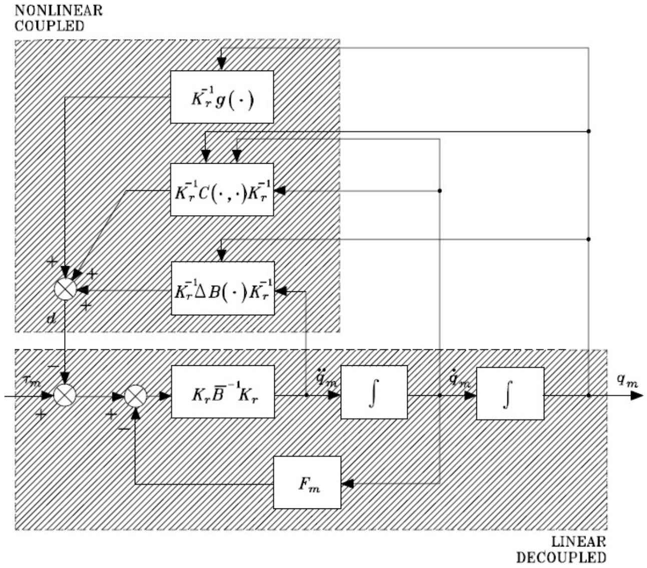

Figure 5: Block Diagram of the Decentralized Control

To actually build this model several approximations had to be made. First the B matrix was considered to be a diagonal matrix composed by by the values of maximum inertia calculated above, presented on table 2. Then we considered the disturbances negligible (making the model linear) and $K_r = 1$.
Using this simplifications we can derive a block diagram for a PD independent joint controller with gravity compensation.
To calculate K_P and K_D we can see in the block diagram that the torque will be given by:

$$\tau = K_P(q_d - q) + K_D\dot{q} + g(q) \tag{3}$$

Which gives the transfer function:

$$G = \frac{\frac{K_P}{B}}{s^2 + \frac{K_D}{B}s + \frac{K_P}{B}} \tag{4}$$

This is a second order system with $2\xi w_n = \frac{K_D}{B}$ and $w_n^2 = \frac{K_P}{B}$. Using $\xi = 1$ (since we want a critically damped system) we can calculate the two gains from the natural frequency.

$$K_P = Bw_n^2 \quad K_P = 2Bw_n \tag{5}$$

The natural frequency can be calculated from the lowest resonance frequency of the manipulator ($w_n = 0.1w_r$). Considering this frequency 20Hz, the natural frequency will be $w_n = 4\pi$ rad/s and the gains will be as seen in table 3

Joint$_i$	K$_P$	K$_D$
1	67.0501	10.6714
2	62.9760	10.0229
3	15.3303	2.4399
4	3.4978	0.5567
5	2.8488	0.4534
6	2.6814	0.4268

Table 3: Gain values for each joint controller

The Simulink model is represented in the appendix B.

5 Centralized Inverse Dynamics Control

The dynamic model of the manipulator can be express as,

$$B(q) \cdot \ddot{q} + C(q, \dot{q}) \cdot \dot{q} + F \cdot \dot{q} + g(q) = \tau \tag{6}$$

where for simplicity it can be rewritten as,

$$B(q) \cdot \ddot{q} + n(q, \dot{q}) = \tau \tag{7}$$

5.1 Design of the controller

Since the dynamic model is already written in 7. Using it as a control law u, this leads to a system described by,

$$\ddot{q} = y$$

$$y = -K_P \cdot q - K_D \cdot \dot{q} + r \tag{8}$$

$$\ddot{\tilde{q}} + K_D \cdot \dot{\tilde{q}} + K_P \cdot \tilde{q} = 0 \tag{9}$$

The resulting block scheme is shown bellow.

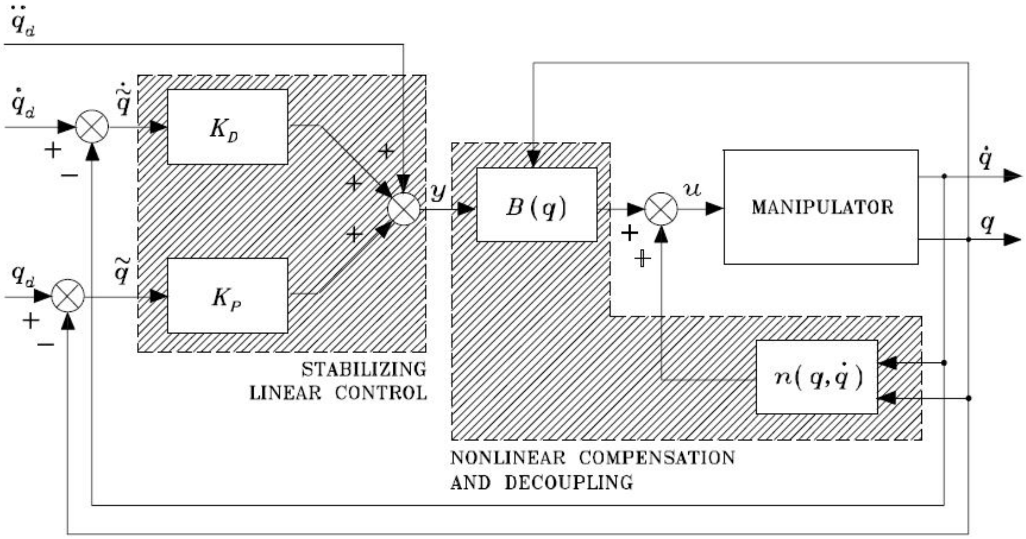

Figure 6: Block Diagram of the Centralized Inverse Dynamics Control

The matrices K$_P$ and K$_D$ are asymptotically stable, under the assumption of positive definite. Considering the matrices as diagonal ones, they give a decoupled system response in each joint. This is a characteristic of second-order system with natural frequency W_{ni} and damping ratio ξ_i for,

$$K_P = diag\{W_{n1}^2, \ldots, W_{nn}^2\} \quad K_D = diag\{2\xi_1 W_{n1}^2, \ldots, 2\xi_n W_{nn}^2\} \tag{10}$$

Using the same assumptions as in the decentralized control, the values obtained are $\xi = 1$ and $W_n = 4\pi\ rad/s$. Implementing this values for the design of the controller by the equation 10, we ended up with the following gain matrices,

$$K_P = 157.9137 \cdot I_6 \tag{11}$$

$$K_D = 25.1327 \cdot I_6 \tag{12}$$

The Simulink model is represented in the appendix D.

6 Results Analysis

In order to evaluate the designed controllers and the compare them we created a trajectory for the end effector, which consisted in performing a capital L followed by a capital C, combining both rectilinear and circular trajectories. The results obtained for both controllers can be seen bellow.

The desired joint trajectories were taken from the CLIK model of the previous work. As this model had already position and orientation errors, these had to be taken out when comparing the controllers errors to have a better idea of their performance.

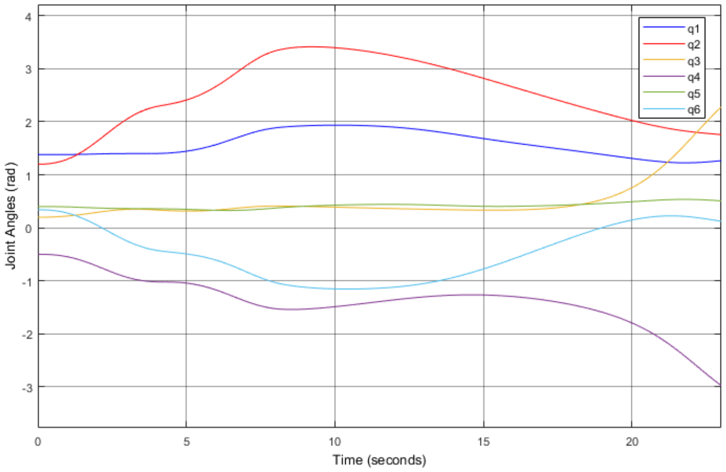

Figure 7: Joint angles

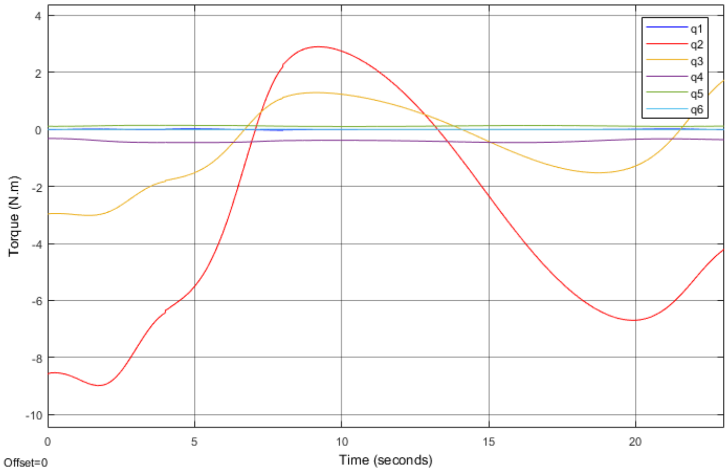

Figure 8: Torque

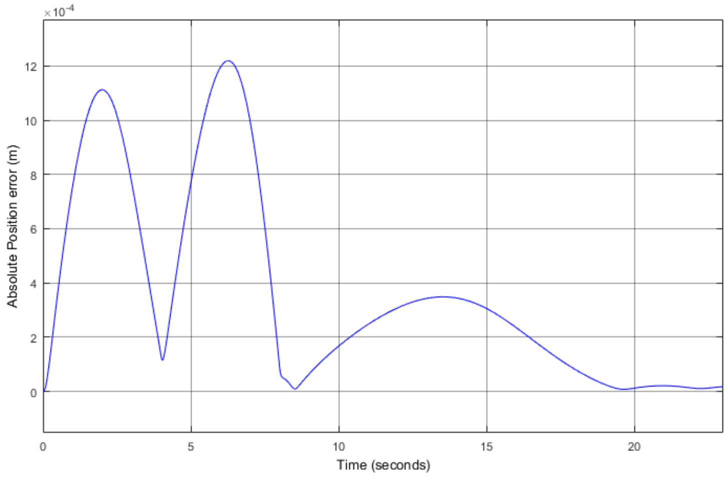

Figure 9: Absolute position error - centralized controller

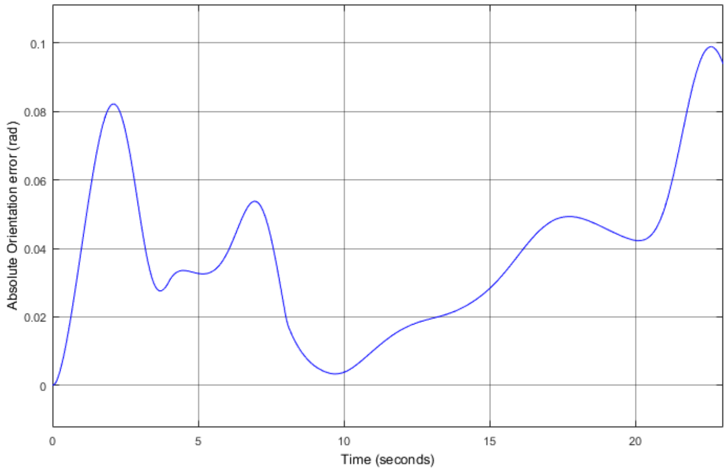

Figure 10: Absolute orientation error - centralized controller

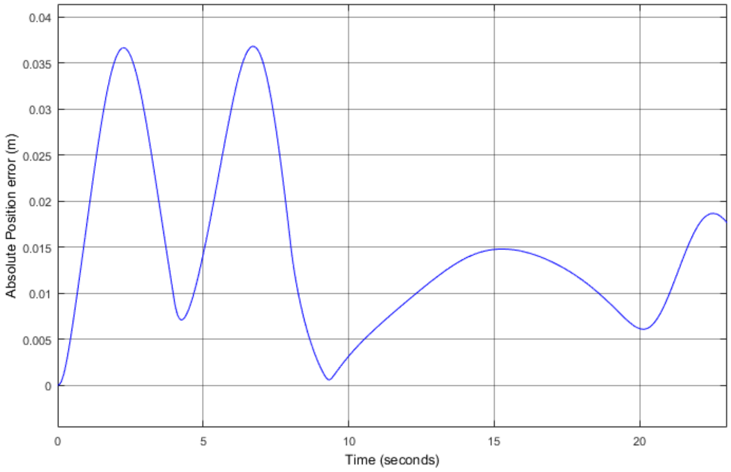

Figure 11: Absolute position error - decentralized controller

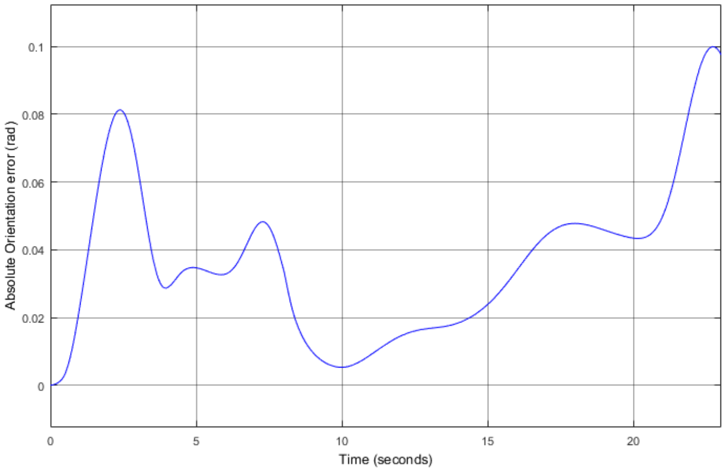

Figure 12: Absolute orientation error - decentralized controller

We can see that both controllers gave similar responses. The maximum error obtained with centralized control was approximately 0.00121 m and 0.097 rad for position and orientation, respectively. With decentralized controller the maximum errors obtained were 0.036 m and 0.1 rad, having a worse position error than the previous and a very similar orientation error.

This result alone would prove a better performance of the centralized controller, even though the decentralized controller is still a viable option. But if we decrease the time to perform the task (increasing the joint velocities) we will see that the decentralized starts to loose credibility.

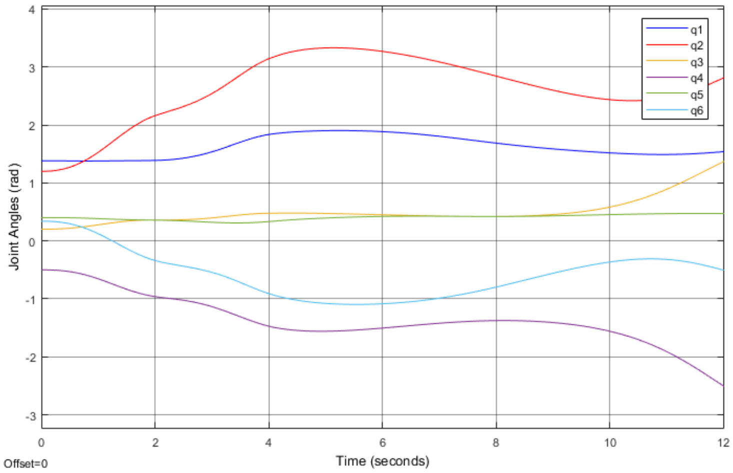

Figure 13: Joint angles for increased speed

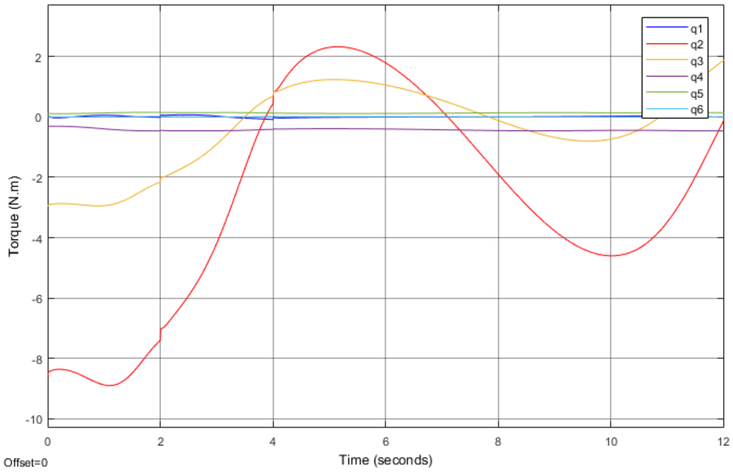

Figure 14: Torque for increased speed

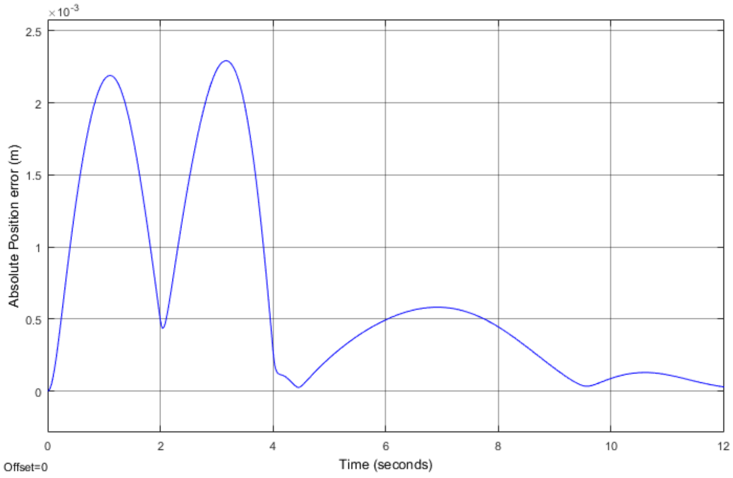

Figure 15: Absolute position error for increased speed - centralized controller

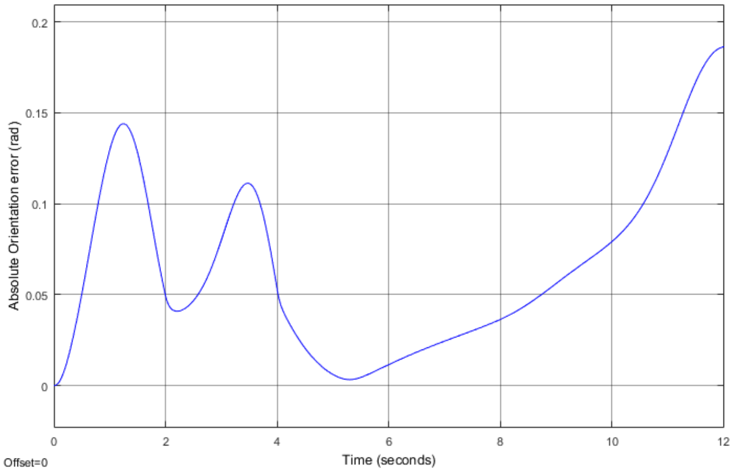

Figure 16: Absolute orientation error for increased speed - centralized controller

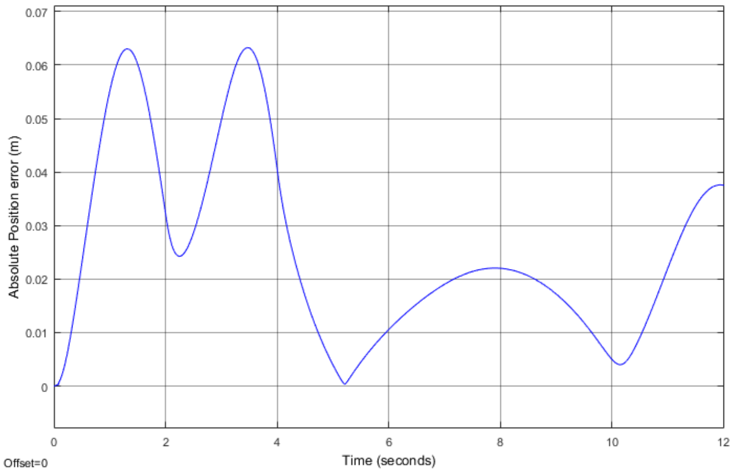

Figure 17: Absolute position error for increased speed - decentralized controller

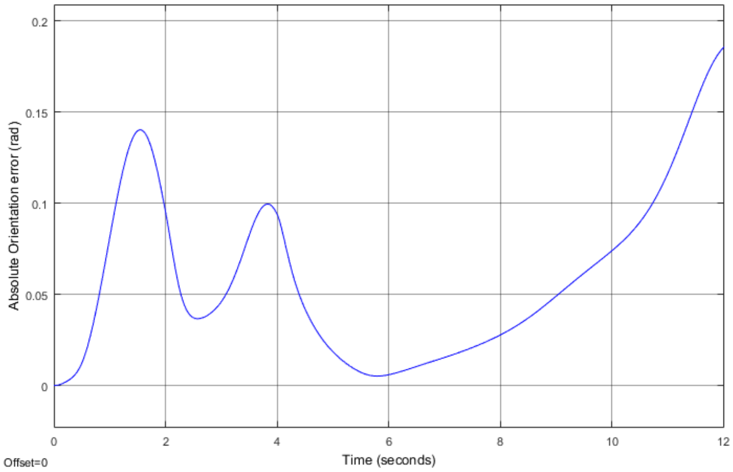

Figure 18: Absolute orientation error for increased speed - decentralized controller

For this case the maximum errors were 0.0023 m and 0.186 rad for the centralized controller and 0.063 m and 0.1859 rad for the decentralized controller. An interesting thing to notice is that as the time was shortened by half the errors is nearly the double as before.

Increasing the speed makes the decentralized controller significantly less acceptable. Although the error also increases in the centralized controller the order of the position error is still in the order of millimeters. To confirm this we made another test shortening the time by half again.

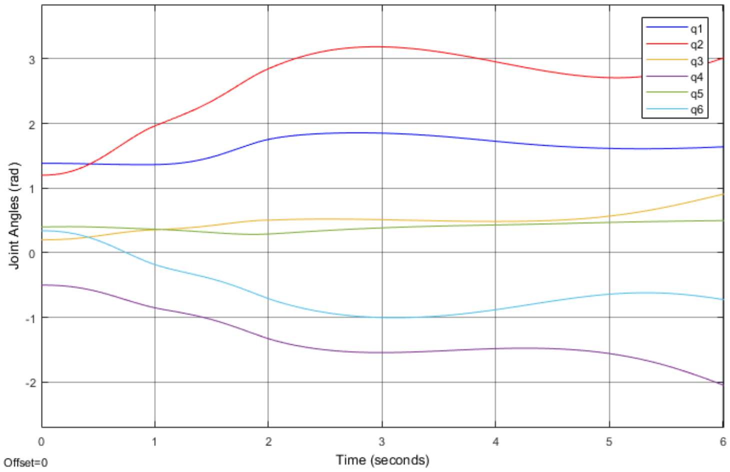

Figure 19: Joint angles for increased speed 2

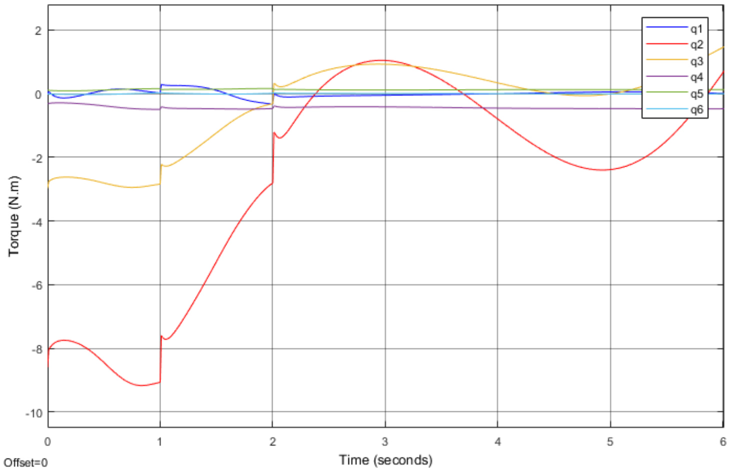

Figure 20: Torque for increased speed 2

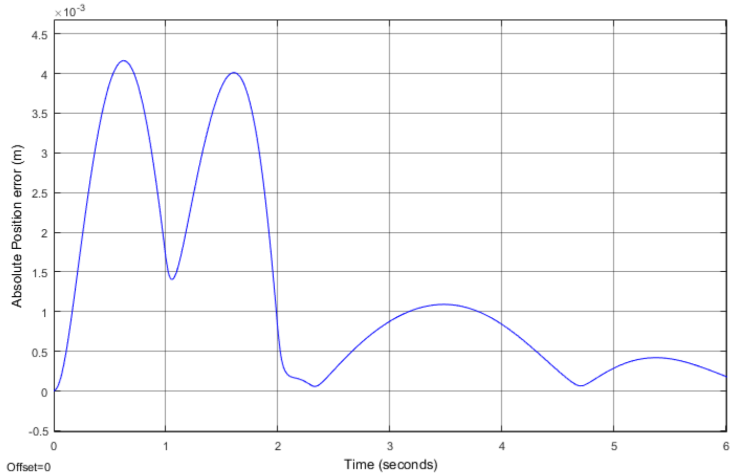

Figure 21: Absolute position error for increased speed 2 - centralized controller

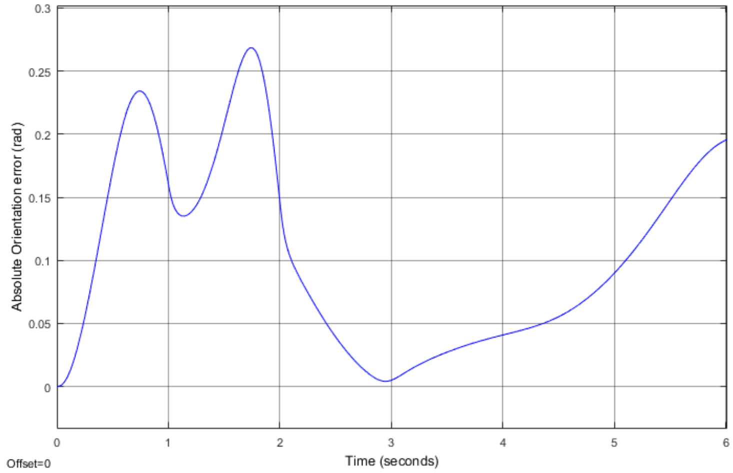

Figure 22: Absolute orientation error for increased speed 2- centralized controller

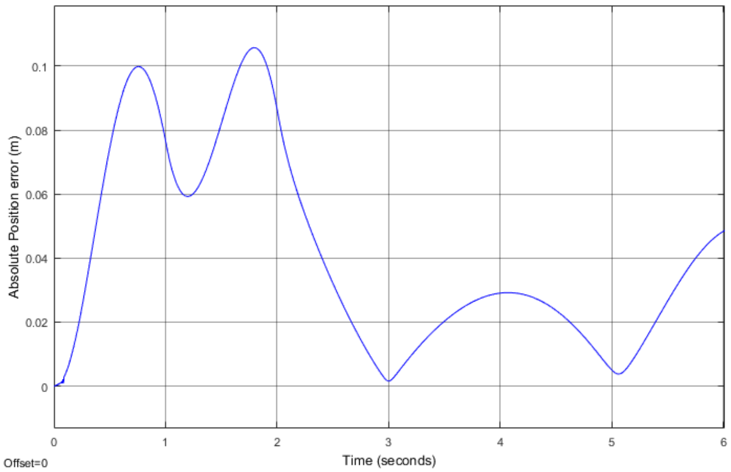

Figure 23: Absolute position error for increased speed 2- decentralized controller

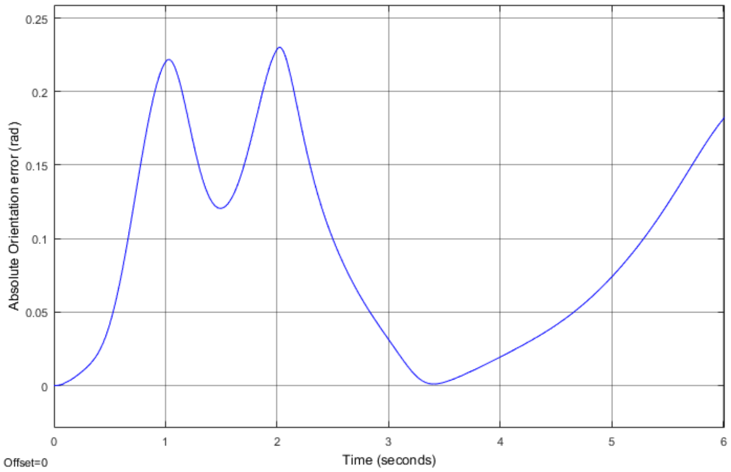

Figure 24: Absolute orientation error for increased speed 2- decentralized controller

To note that at this speed the maximum orientation error came from the linear trajectories and not the circular one. We can also see the shape circular trajectory flattening as speed increases for both controllers. Also note that the orientation error was almost the same for the both controllers.

7 Conclusion

As this report demonstrates, the dynamic model, with the Newton−Euler formulation, was successfully built and validated by two tests. One with the manipulator arm at rest and another by making the manipulator arm behave as a gravitational pendulum.

Regarding the controllers, both were successfully designed and implemented. The decentralized controller wasn't as successful as the centralized one for high velocity trajectories and also as a higher position and pose errors, but more significant in the position one. From this, the best controller is the centralized one.

References

[1] $www.universal - robots.com$

[2] $www.universal{-}robots.com/media/207442/ur3usermanualenglobal.pdf$

[3] B. Siciliano, L. Sciavicco, L. Villani, G. Oriolo (2009) *Robotics: Modelling, Planning and Control*. Springer-Verlag London, 1st edition.

A Manipulator Links Inertia Tensors

The Inertia Tensors units are $[Kg.m^2]$.

$$I_1 = \begin{bmatrix} 0.0011 & 0 & 0 \\ 0 & 0.0009 & 0.0073 \\ 0 & 0.0073 & 0.0009 \end{bmatrix}$$

$$I_2 = \begin{bmatrix} 0.0021 & 0 & 0.0003 \\ 0 & 0.015 & 0 \\ 0.0003 & 0 & 0.015 \end{bmatrix}$$

$$I_3 = \begin{bmatrix} 0.0012 & 0 & -0.0004 \\ 0 & 0.0085 & 0 \\ -0.0004 & 0 & 0.0083 \end{bmatrix}$$

$$I_4 = \begin{bmatrix} 0.2938 & 0 & 0 \\ 0 & 0.2610 & -0.0199 \\ 0 & -0.0199 & 0.2318 \end{bmatrix} * 10^{-3}$$

$$I_5 = \begin{bmatrix} 0.3462 & 0 & 0 \\ 0 & 0.3127 & 0.0267 \\ 0 & 0.0267 & 0.2449 \end{bmatrix} * 10^{-}3$$

$$I_6 = \begin{bmatrix} 0.5678 & 0 & 0 \\ 0 & 0.5678 & 0 \\ 0 & 0 & 0.8652 \end{bmatrix} * 10^{-}4$$

B Decentralized Simulink Model

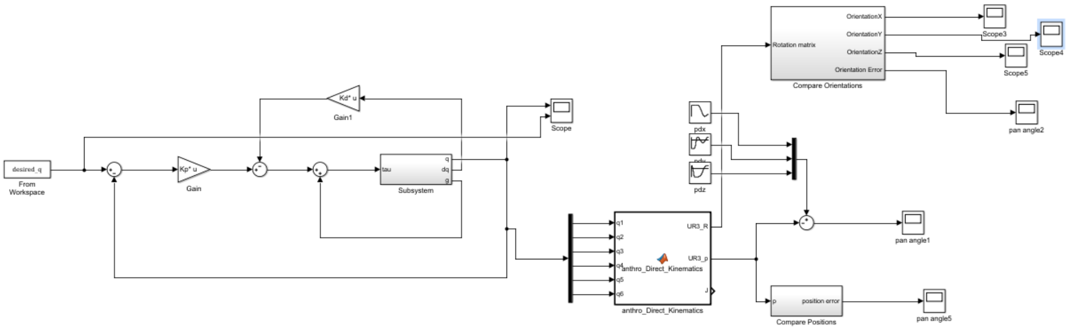

Figure 25: Decentralized control

C Manipulator Simulink Model Subsystem

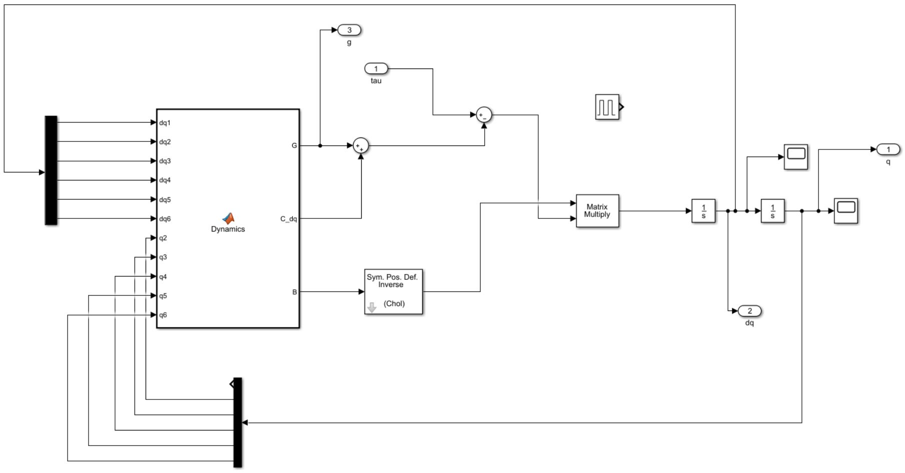

Figure 26: Manipulator Subsystem

D Centralized Simulink Model

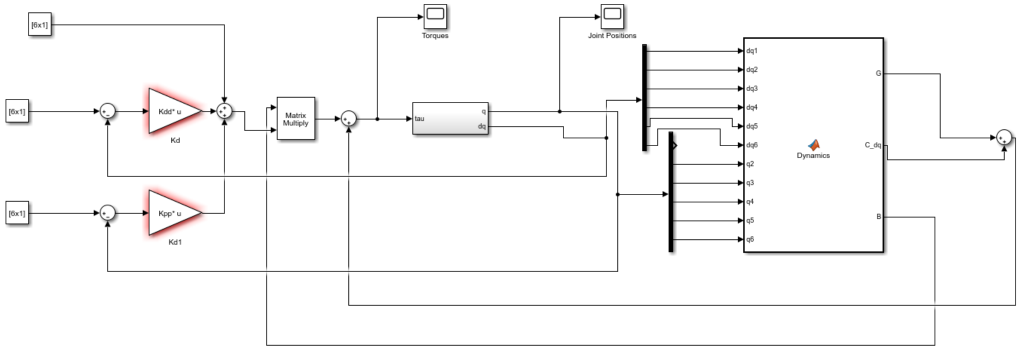

Figure 27: Centralized Control