UNIVERSITÀ DI BOLOGNA

School of Engineering

Master Degree in Automation Engineering

MODELING AND SIMULATION OF MECHATRONIC SYSTEMS M

Modelling and Control Simulation of UR5

With Harmonic Drive

Professor: **Alessandro Macchelli**

Students:
Nzangue Ange derick
Mahdi Hamadi
Foasse Nyie Richard Duval
Zijian Jiang

Academic year 2022/2023

Abstract

UR robotic arms are from a series of lightweight, fast, easy to program, flexible, and safe robotic arms with 6 degrees of freedom. The fairly open control structure and low level programming access with high control bandwidth have made them of interest for many researchers. This paper presents a complete set of mathematical kinematic and dynamic, Matlab, and Simscape model for the UR5 robot. In the following project, we present a complete implementation and the dynamical model of the UR5 manipulator, using Matlab, and Simscape models to control the UR5 robot. The accuracy of the developed mathematical models are demonstrated through kinematic and dynamic analysis. The Simscape model is developed based on these models to provide high quality visualisation of this robot for simulation of it in Matlab environment.A position control system has been developed to demonstrate the use of the models and for cross validation purpose.

Contents

Introduction

The UR5, developed by Universal Robots, represents a new era of collaborative robotic solutions. It's a lightweight, six-axis robotic arm that combines a compact design with powerful capabilities. Designed to work alongside humans safely, the UR5 is suitable for a wide range of applications, including manufacturing, logistics, research, and development. At the core of the UR5's remarkable performance lies its harmonic drive, a crucial component that enables its precision and agility. The harmonic drive is a type of gear mechanism known for its compact size, high reduction ratio, and exceptional torque-to-weight ratio. This technology is pivotal in defining the behavior of the UR5 robot. In the present paper, a thorough mathematical model for kinematics and dynamics of the UR5 robot is presented. The kinematic model includes full mathematical development for the forward and inverse kinematic equations of the robot. The dynamic model gives the equation of motion of the robot and access to the parameters of that equation including the mass inertial matrix, centrifugal and Coriolis matrix and gravity force vector. That will help on making the control action so that the manipulator is programmed to track a time varying joint trajectory specified to accomplish a well-defined task.

Chapter 1

Definition of the dynamic model

Figure 1.1: Universal Robots UR5

Universal robot (UR5) is consided as a collaborative robot since is designed to collaborate with humans. it's safe to work side by side with humans in a co-working space as show in the figure 1.2.Figur 1 show the universal robot type UR5 with 6 Degree of freedom (Dof) profuct by Universal Robots Company.

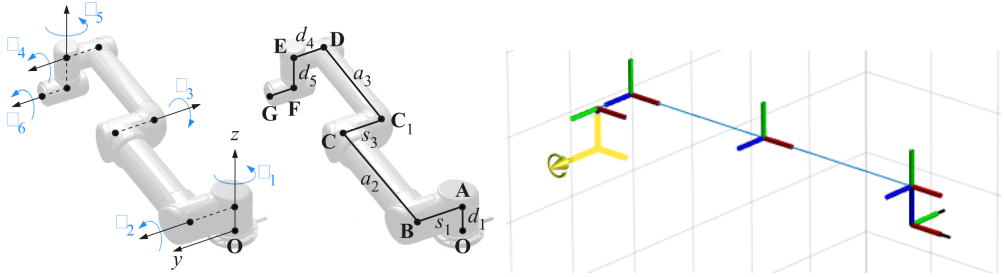


Figure 1.2: Universal Robots schematics and frameworks

1.1 Parameters for the dynamic model

Dynamic is the study of how forces and torques affect the motion.Mathematically,this relationship is given through the equation of motion in the following form.

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

with description, $M(q) \in R^{6*6}$ is the inertia matrix, $C(q, \dot{q}) * \dot{q} \in R^{6*6}$, is the coriolis matrix and the centrifugal acceleration, $g(q) \in R^6$, is the gravity vector. $\tau \in R^6$, is the torque input vector,$q_i \in R$, is the joint angle of the robot at the $\imath$ joint $\dot{q}_i \in R$, is the angular velocity of the robot at the $\imath$ joint $\ddot{q}_i \in R$, is the angular acceleration of the robot at the joint $\imath$.

The figure 1.3 shows the specifications of the UR5 and the the figure 1.4 is a DH (Denavit-Hartenberg) parameterd that defines the parameters for the frame of reference in each link attached to the robot with q_i is the angle of x_{i-1} to x_i measured to the z_i, d_i is the distance from x_{i-1} to x_i measured to the z_i, a_i is the distance from z_{i-1} to z_i measured to the x_i α_i is the angle of z_i to z_{i+1} measured to the x_i.

weight	18.4 kg
payload	5 kg
reach	850.0 mm
joint ranges	± 360 °
joint max speed	180 °/s
TCP max speed	1 m/s
degrees of freedom	6 rotational joints
repeatability	± 0.1 mm
I/O power supply	12 V/24 V, 600 mA
communication	TCP/IP, Ethernet socket & Modbus TCP
programming	Polyscope graphical user interface
IP classification	IP54
power consumption	150 W
power supply	10-240 VAC, 50-60 Hz
materials	aluminium, ABS and plastic
temperature	Working range of 0-50 °C
operating life	35000 h

Figure 1.3: Technical specifications of the UR5 robotic arm.

UR5 Robot Arm:: 6 axis, RRRRRR, modDH, slowRNE

j	theta	d	a	alpha	offset
1	q1	0.089159	0	1.5708	0
2	q2	0	-0.425	0	0
3	q3	0	-0.39225	0	0
4	q4	0.10915	0	1.5708	0
5	q5	0.10915	0	-1.5708	0
6	q6	0.0823	0	0	0

Figure 1.4: UR5 Denavit-Hartenberg parameters.

1.2 SimMechanics model development for the UR5 robot

For developing Simmechanics model of a robotic system knowing the physical parameters and structural properties of the robot is essential. At the first step, coordination, orientation and dimension of each parts of the robot are required. DH parameters are considered to provide these geometrical specifications of the robot. The second step is to assign mass, centre of mass and inertia tensor, for each part of the robot. It is noted that in the Simmechanics models, the COMs should be expressed in a frame located at the centre of geometry (CoG) of the body. This is different with the CoM obtained by DH parameters, because the CoM based on the DH parameters are commonly expressed in the joint frames. Therefore, for the Simemchanics model development, transformation of the COMs from the joint frame to the frame located at the CoG of the bodies is required. After this transformation and using the concept of multi-body systems modelling of Simmechanic environment a model was developed for the UR5 robot. Fig 1.5 shows the Simmechanics blocks used for modelling of the UR5 manipulator with a controller for it. This figure shows the Controller and Robot parts. The Robot part contains of three types of blocks including Base, Joint and Link blocks. The Controller part is a Matlab function (code) and is not part of Simmechanics model of the UR5. Details of the block of the Robot part are explained in below

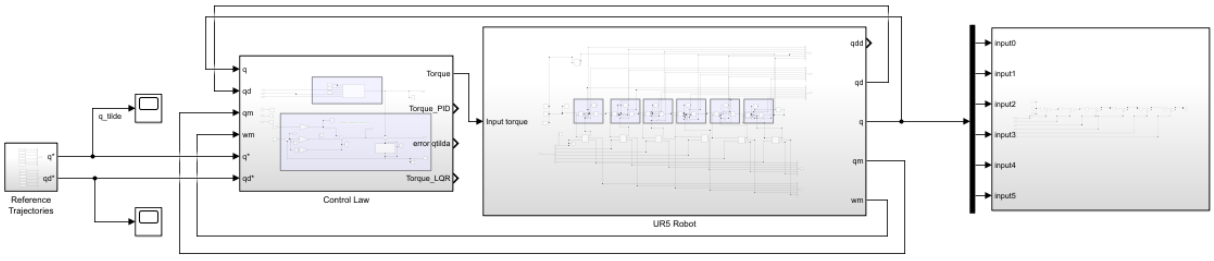
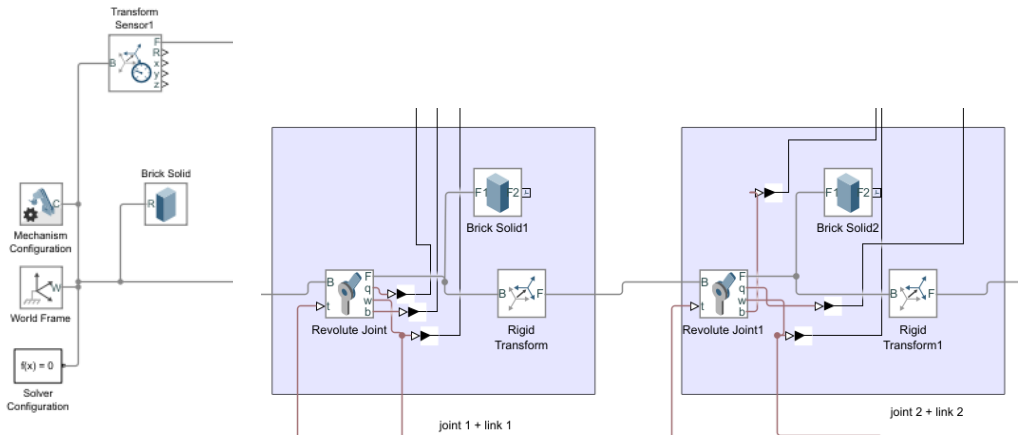


Figure 1.5: UR5 Simmechanics blocks in Matlab Simulink.

1.2.1 Base block

For any Simmechanics model, it is needed to use three basic blocks that are Solver, World frame and Mechanism configuration blocks. These blocks define the system' s environment (world) and its properties, such as gravity constant and direction. The base of the robot should be defined immediately after the block World frame. These blocks are depicted in Fig 1.6(a). The block with the name of Solid in this figure is a representative of the base of the robot. For the Base block, there is an output port Joint1, this port is for the physical connection between the base of the robot to the next block.



(a) The robot's Base block details.

(b) The robot's Joint and link blocks details.

Figure 1.6: Base block

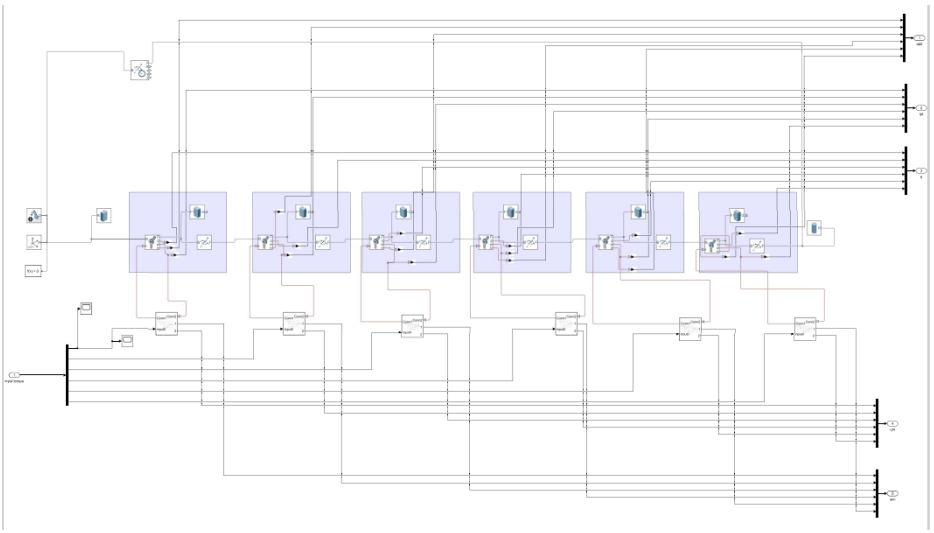


Figure 1.7: The robot block details.

1.2.2 Joint block

There are several Joint blocks that are shown in Fig. 1.7 to represent the mechanism of the joints between the robot links. Although these blocks are similar, they have different sets of parameter values. The details of the Joint1 block is illustrated in Fig. 1.6(b). For the Joint blocks, there are two inputs, one for Simmechanics physical connection between parts of the mechanism, which is named B and the other one for the torque signal that is applied to the joint refereed as τ_1 (t) in Fig.1.6(b). There are also three measurement output signals: q_1 (q), dq_1 (w) and ddq_1 (b) that are for the joint angle, velocity and acceleration. These signals are the system's states of the joints and they are useful for control purposes. In the Joint block, the subblock Revolute Joint1 is a Simmechanics block of a revolute joint with its corresponding actuator.
The input of this subblock is a torque signal that is applied by the control system (here is PID +

gravity compensation). The output of the Revolute Joint1 block are the joint angle, joint velocity and joint acceleration measurement signals with dimension of radian (rad), radian per second (rad/s) ans radian per second squart (rad/s^2), respectively. Finally, the output of the whole Revolute Joint1 block is a Simmechanics connection with the name of Link1 that goes to the Link1 block.

1.2.3 Link Block

There are also several Link blocks in Fig.1.7 to represent the robot links. The details of the Link1 block is demonstrated also in Fig.1.6(b). These blocks also have similar structure, hence only Link1 is explained here. These blocks contain pairs. Each pair is consisted of a Rigid Transformation block and a Solid block. The most important pair is the Solid block part of it, this is because it creates a link body that contains the inertia properties of that link. These Link blocks are connected to a joint from its input and a joint from its output, except the last Link block. For the Link1, the input of the Link1 is from the Joint1 and the output of it is applied to the Joint2 as shown in fig.

1.2.4 Simemchanics model parameters

There are several reports for the parameters of the COMs e.g., [1], [2], [3], one of them is from the manufacturer of the robot. For the Simmechanics model presented here we used the manufacturer parameter. Other specifications, like DH parameters and inertia properties are from [3]. These parameters are used for the model and the Simmechanics model is fully obtained. The run of the Simemchanics model creates the results shown in Fig.1.8

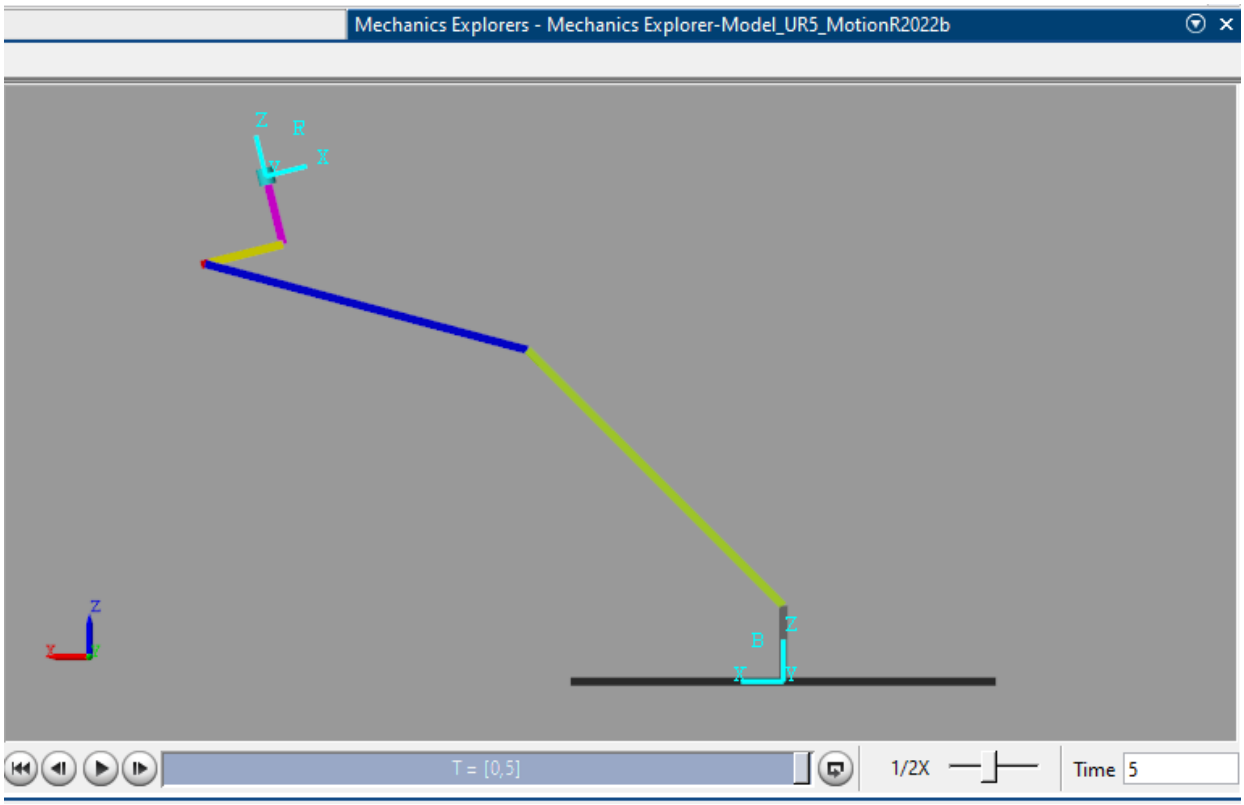


Figure 1.8: Graphical result of UR5 modelled in Simmechanics.

1.3 SimMechanics model development for the Harmonic drive

Implementing correct robot drive systems is of paramount importance for achieving smooth low-speed motion and compliant behaviour in contact situations. The transmission elements of robot drive system should have important features, such as zero backlash, low weight, low friction losses, compact size and high torque capacity. Harmonic drive contains these properties more than other drive systems,because of its unconventional gear-tooth meshing action.

In order to implement the model of harmonic drive in Simscape, simplified bondgraph of the system is given in fig 1.9:

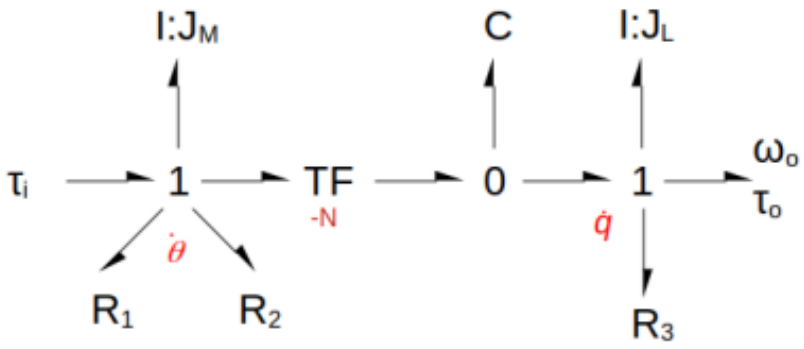


Figure 1.9: Harmonic Drive bond graph.

In UR5 robot manipulator two different harmonic drive modules have been utilized, namely for the first three joints HFUS20 and for the last three joints HFUS14 with gear ration of 100. Based on the bond graph given in figure 1.9, the harmonic drive can be modelled as in figure 1.10. This model has been created for each joint of the robot manipulator and contains : Friction, Inertia, Stiffness, Input Torque from the controller, Sensing of position and velocity of the motor.

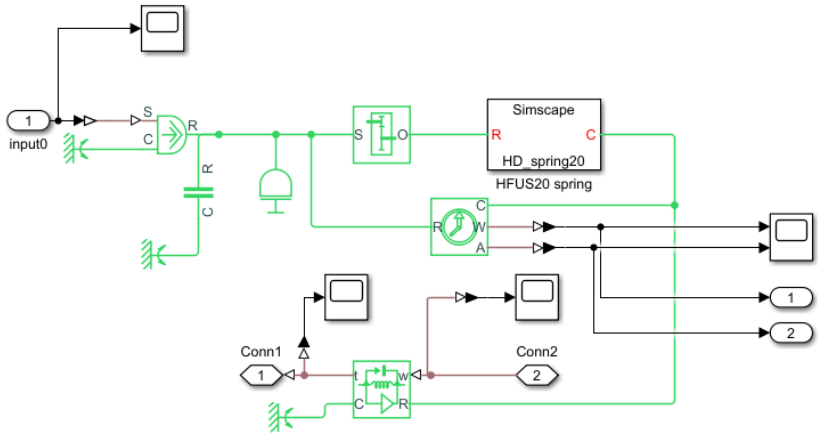


Figure 1.10: Harmonic Drive model in Simscape Multibody (formerly SimMechanics).

1.3.1 Nonlinear Spring Element

The key element in harmonic drive model is nonlinear spring element which represents flexspline elasticity.

Is modelled by a custom Simscape block. The custom model includes a specific structure Matlab-based, with a coding part. The torsional stiffness containt in the non-linear charaterisctic can be evaluated based on the below torque-torsion curve.

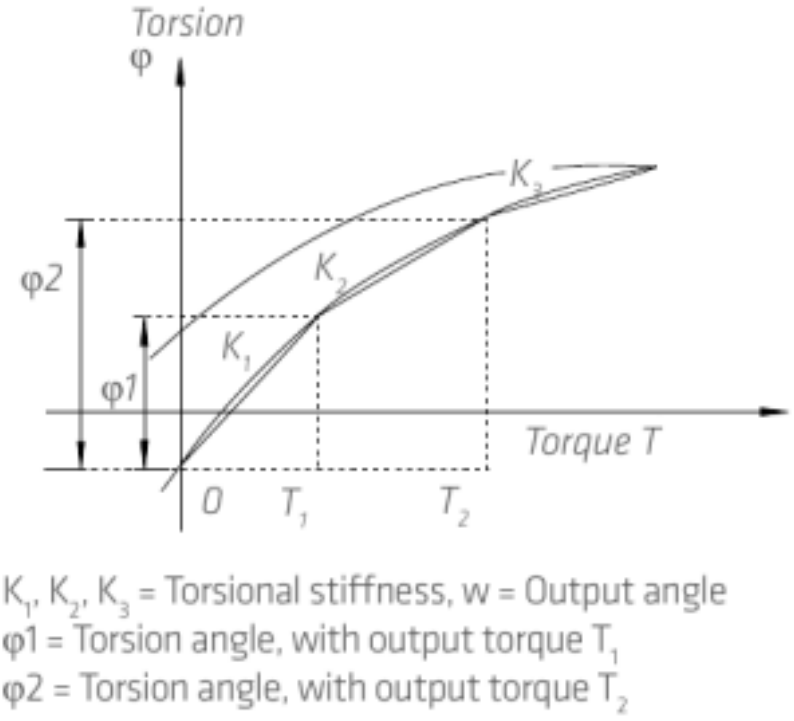


Figure 1.11: Torque-torsion curve.

It is obvious from the figure 1.11 that the curve consist of three regions, namely low torque region, middle torque region and high torque region. The torsional stiffness for each region can be calculated by firstly linearizing the overall curve and calculating stiffness values as slope of these curves. So, torsion angle for each region can be calculated then as:

$$\begin{cases} \varphi = \frac{T}{K_1} & : \ T_1 \geq T \\ \varphi = \frac{T}{K_1} + \frac{T-T_1}{K_2} & : \ T_1 < T \leq T_2 \\ \varphi = \frac{T}{K_1} + \frac{T_2-T_1}{K_2} + \frac{T-T_2}{K_3} & : \ T > T_2 \end{cases}$$

However, the input for the spring element is the torsion angle which is obtained by integrating the angular velocity of the flexspline and the output is the torque. So, the formulas have to be reversed. Namely, firstly torsion angles corresponding to torques T_1 and T_2 are calculated which can be called φ_1 and φ_2 respectively.

$$\begin{cases} \varphi_1 = \frac{T}{K_1} \\ \varphi_2 = \frac{T}{K_1} + \frac{T_2-T_1}{K_2} \end{cases}$$

Then, the input torsion angle is compared in order to determine the region of activity and so, the corresponding torsional stiffness value. And at the end, output torques are calculated based on determined torsional stiffness values as below:

$$\begin{cases} \varphi_1 \leq \varphi \Rightarrow T = \varphi * K_1 \\ \varphi_1 < \varphi \leq \varphi_2 \Rightarrow (\varphi - \varphi_1) * K_2 + T_1 \\ \varphi > \varphi_2 \Rightarrow T = (\varphi - \varphi_2) * K_3 + T_2 \end{cases} \tag{1.1}$$

1.3.2 Datasheet Parameters

As mentioned earlier, our manipulator is equipped with two distinct types of harmonic drive modules. The first three joints are equipped with HFUS20 modules, while the last three joints utilize HFUS14 modules. We've extracted relevant parameters from the datasheet:

		HFUS-20	**HFUS-14**
Reduction Ratio		100	100
Friction		6.4×10^4	6.4×10^4
Inertia		0.404×10^4	0.091×10^4
Limit Torques	T_1	7	2
	T_2	25	6.9
Torsional Stiffness	K_1	16×10^3	4.7×10^3
	K_2	25×10^3	6.1×10^3
	K_3	29×10^3	7.1×10^3

Table 1.1: Harmonic Drive HFUS Datasheet Parameter

Chapter 2

Control Strategy

2.1 PID controller

Generally, the task of the robot arm includes tracking the input trajectory as smooth and precise as possible. So, it is desired to find the structure of the controller which ensures global asymptotic stability of the posture.

The ideal behaviour of the robot would occur when the angle error is lower and lower until it becomes null. This task is achieved by the PID controller, which is implemented for ensuring a good performance of the robot work. That means getting the desired angle as fast as possible with a null error and a great stability around the equilibrium point. The state of the system is given by below vector:

$$x = [\tilde{q}^T, \dot{q}^T]^T \ , \quad \tilde{q} = q_d - q \tag{2.1}$$

The candidate Lyapunov function can be choosen as:

$$V(\dot{q}, \tilde{q}) = \frac{1}{2}\dot{q}^T B(q)\dot{q} + \frac{1}{2}K_p\tilde{q} > 0, \forall \tilde{q}^T, \dot{q}^T \neq 0 \tag{2.2}$$

where Kp is (nxn) symmetric positive definite matrix. In order to obtain stability, the derivative of Lyapunov function has to be made nonpositive. So, firstly derivative of the Lyapunov function has to be obtained which is given below:

$$\dot{V}(\dot{q}, \tilde{q}) = \frac{1}{2}\dot{q}^T B(q) - 2C(q, \dot{q})\dot{q} - \dot{q}^T F\dot{q} + \dot{q}^T(u - g(q) - K_p\tilde{q}) \tag{2.3}$$

The first term of the equation is zero due to Christoffel symbols:

$$N = \dot{B} - 2C \tag{2.4}$$

The second term is negative definite. So, input has to be chosen as:

$$u = g(q) + K_p\tilde{q} \tag{2.5}$$

To get negative semi-definite $\dot{V}$. Additional damping term Kd can be added to the control law as given below:

$$u = g(q) + K_p\tilde{q} + K_i\int_0^t \tilde{q}(t)dt - K_d\dot{q} \tag{2.6}$$

to improve system time response, where Kd is positive definite matrix. Overall the control law corresponds to nonlinear compensation action of gravitational terms with a linear proportional-derivative action.

2.2 Gravity Compensation

In order to realize PID and gravity compensation controller, gravity compensator has to be implemented. The essential part of the controller is gravity compensator, because gravity term is generally the dominant term in robot dynamics. It presents even when the robot is stationary or moving slowly. To calculate the gravity term, potential energy for each link has to be calculated which is configuration dependent. Below the formula illustrates the calculation of potential energy for each link:

$$P_i = m_i g^T r_{ci} \tag{2.7}$$

where r_{ci} is the coordinate of the center of mass of link i. After getting each link's potential energy, the overall potential energy of the system has to be calculated by summing the individual potential energies which is given below (ignoring robot elasticity):

$$p = \sum_{i=1}^{n} P_i = \sum_{i=1}^{n} m_i g^T r_{ci} \tag{2.8}$$

To obtain the gravity term for each of the manipulator's link, the partial derivative of the total potential energy has to be calculated with respect to each of the joint generalized coordinates as the following:

$$g_k = \frac{\partial P}{\partial q_k} \tag{2.9}$$

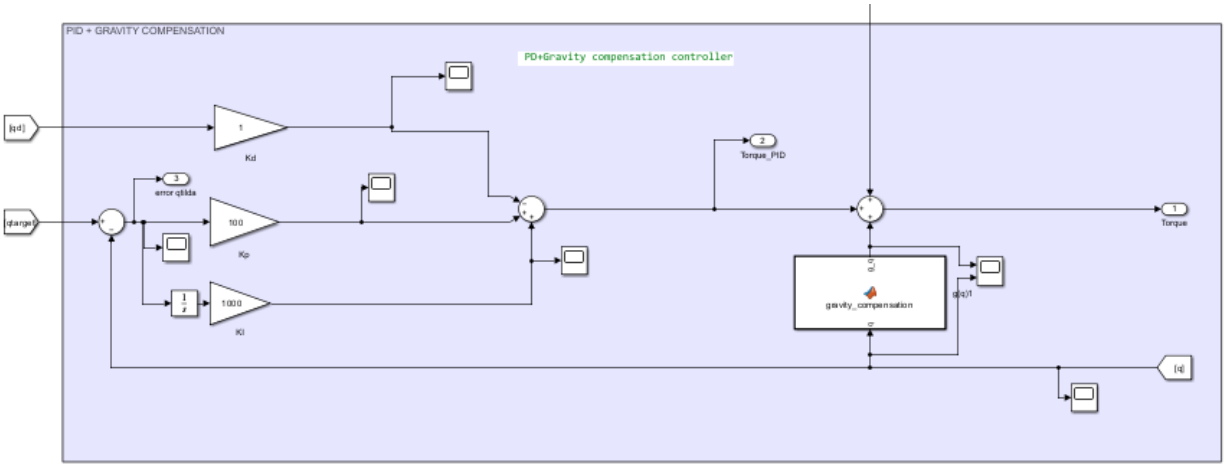


Figure 2.1: PID + Gravity Compensation Controller Scheme.

2.3 Tracking Trajectory (LQR Controller)

In the previous sections PID and gravity compensation controller has been used to control the robot manipulator while tracking the trajectory. However, after adding the harmonic drive

model into the overall robot model, it should also be taken into account in order to track the given trajectory successfully.

In this part, LQR controller will be applied to get the desired motion. However, in order to construct the LQR controller, the differential equations that represent the system has to be obtained. In order to get these equations, bond graph in figure 1.9 has to be assigned with correct causalities. The graph with causalities assigned is given in figure 2.2

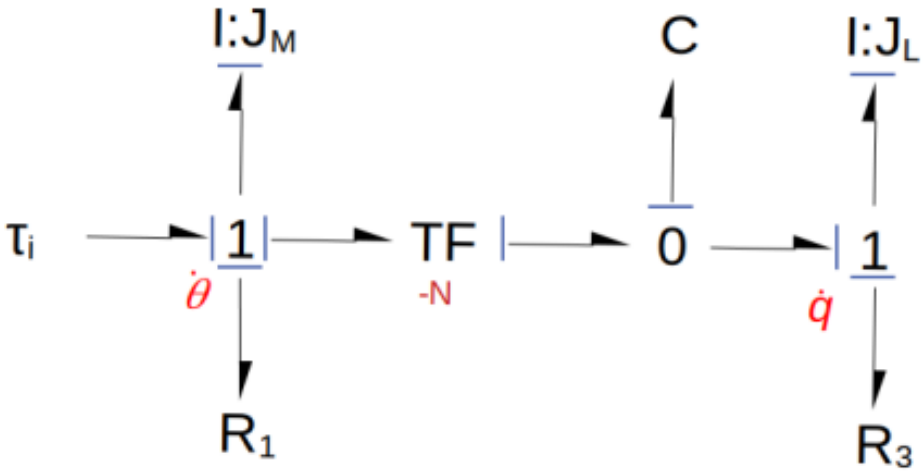


Figure 2.2: Bond graph with causalities.

After assigning causalities, it can be noted that there is neither algebraic loop nor derivative causality. So, the state vector is $x = \begin{bmatrix} p_m \\ p_l \\ \phi \end{bmatrix}$

After state vector has been determined, bond graph equations can be written in terms of state elements.

$$\dot{p}_m = \tau_i - \frac{K\phi}{N} - \frac{Bp_m}{J_m} \tag{2.10}$$

$$\dot{\phi} = -\frac{p_l}{J_l} + \frac{p_m}{J_m N} \tag{2.11}$$

$$\dot{p}_l = K\phi - \frac{Bp_l}{J_l} \tag{2.12}$$

ϕ can be obtained from equation 2.11 by integrating both sides. If the obtained value of ϕ is plugged into equations 2.10 and 2.12, then below differential equations can be obtained, which represent the system under consideration.

$$J_m\ddot{\theta} + B\dot{\theta} + \frac{K}{N}(\frac{\theta}{N} - q) = \tau \tag{2.13}$$

$$J_l\ddot{q} + B\dot{q} + K(q - \frac{\theta}{N}) = 0 \tag{2.14}$$

where : J_m and J_l is the inertia of the motor and load sides, τ_i: the input torque, N: the gear ration, B: the damping coefficient, K: the spring constant, $\dot{\theta}$: the motor side angular velocity, $\dot{q}$: the load side angular velocity.

As the LQR controller is designed based on linear model, nonlinearities due to the damping and spring elements have been ignored. In the next step, state-space model of the system has to be obtained. In order to do that the system equations can be rewritten by choosing the state variables as:

$$\begin{array}{ll} x_1 = q & , x_3 = \theta \\ x_2 = \dot{q} & , x_4 = \dot{\theta} \end{array} \tag{2.15}$$

So the system equations become:

$$\begin{array}{rcl} \dot{x}_1 & = & x_2 \\ \dot{x}_2 & = & -\frac{B}{J_l}x_2 - \frac{K}{J_l}x_1 + \frac{K}{J_l N}x_3 \\ \dot{x}_3 & = & x_4 \\ \dot{x}_4 & = & \frac{\tau}{J_m} - \frac{B}{J_m}x_4 - \frac{K}{J_m N^2}x_3 + \frac{K}{J_m N}x_1 \end{array} \tag{2.16}$$

These equations can be grouped in matrix form as below:

$$x = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{K}{J_l} & -\frac{B}{J_l} & \frac{K}{J_l N} & 0 \\ 0 & 0 & 0 & 1 \\ \frac{K}{J_m N} & 0 & -\frac{K}{J_m N^2} & -\frac{B}{J_m} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{1}{J_m} \end{bmatrix} [\tau] \tag{2.17}$$

$$y = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \tag{2.18}$$

From the above matrix form equations, the state-space matrices (A, B and C) can be extracted easily. After the state-space matrices have been obtained, **a linear state feedback controller** can be written as:

$$u_{(t)} = -K^T x + u_{ref} \tag{2.19}$$

Here, u_{ref} is the reference input which is the output from PID and gravity compensation controller and K is the optimal feedback gain which is obtained by solving algebraic Riccatti equation. Schematic of the overall controller is illustrated in figure 2.3.

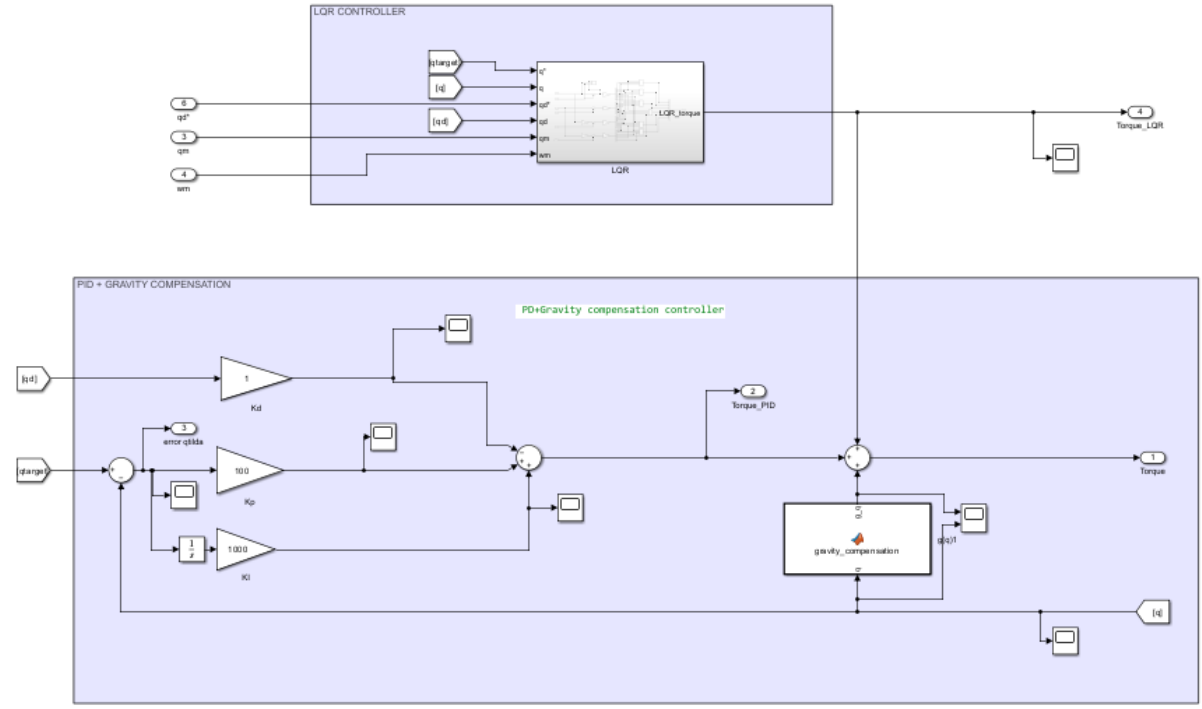


Figure 2.3: PID + gravity compensation and LQR controller.

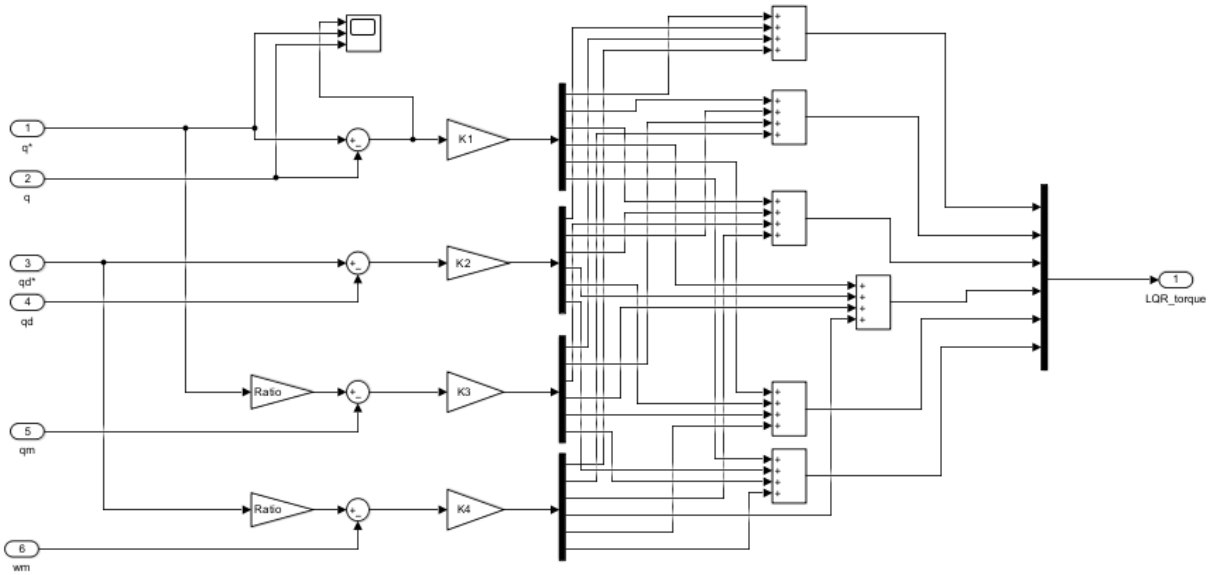


Figure 2.4: Inside the LQR controller.

Chapter 3

Trajectory Planning

3.1 Polynomial Trajectory

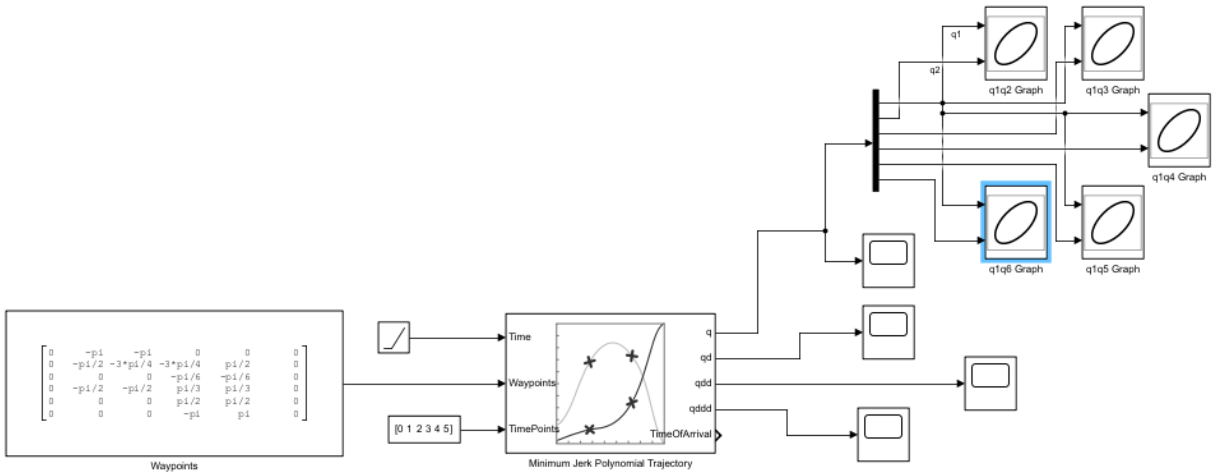


Figure 3.1: the Minimum Jerk Polynomial Trajectory Scheme.

Trajectory planning consists in finding a time series of successive joint angles that allows moving a robot from a starting configuration towards a goal configuration, in order to achieve a task. This trajectory must respect given constraints: for instance, the robot should not collide with the environment; the joint angles, velocities, accelerations, or torques should be within specified limits.If several trajectories are possible, one should choose the one that optimizes a certain objective, such as the trajectory execution time or energy consumption.

Path planning and trajectory planning are crucial issues in the field of Robotics and, more generally, in the field of Automation. Indeed, the trend for robots and automatic machines is to operate at increasingly high speed, in order to achieve shorter production times. The high operating speed may hinder the accuracy and repeatability of the robot motion, since extreme performances are required from the actuators and the control system. Therefore, particular care should be put in generating a trajectory that could be executed at high speed, but at the same time harmless for the robot, in terms of avoiding excessive accelerations of the actuators and vibrations of the mechanical structure. Such a trajectory is defined as smooth. For such reasons, path planning and trajectory planning algorithms assume an increasing significance in robotics.

Path planning algorithms generate a geometric path, from an initial to a final point, passing through pre-defined via-points, either in the joint space or in the operating space of the robot, while trajectory planning algorithms take a given geometric path and endow it with the time information.

Trajectory planning algorithms are crucial in Robotics, because defining the times of passage at the via-points influences not only the kinematic properties of the motion, but also the dynamic ones. Namely, the inertial forces (and torques), to which the robot is subjected, depend on the accelerations along the trajectory, while the vibrations of its mechanical structure are basically determined by the values of the jerk (i.e. the derivative of the acceleration). Path planning algorithms are usually divided according to the methodologies used to generate the geometric path, namely:roadmap techniques, cell decomposition algorithms, artificial potential methods.The algorithms for trajectory planning are usually named by the function that is optimized, namely:minimum time, minimum energy, minimum jerk.

3.2 Trajectory Simulation

In this section we will use the matlab function name **trapveltraj** to generate a trajectory through a given set of input waypoints that follow a trapezoidal velocity profile. The function outputs positions, velocities, and accelerations at the given time samples, tSamples, based on the specified number of samples, numSamples. The function also returns the piecewise polynomial form of the polynomial trajectory with respect to time. we define target position vectors for each joint as:

$$\begin{aligned}
Wp_1 &= \begin{bmatrix} 0, & -\pi, & -\pi, & 0, & 0 \end{bmatrix} \\
Wp_2 &= \begin{bmatrix} 0, & -\pi/2, & -\pi/2, & -3\pi/2, & -3\pi/4 \end{bmatrix} \\
Wp_3 &= \begin{bmatrix} 0, & 0, & 0, & -\pi/6, & -\pi/6 \end{bmatrix} \\
Wp_4 &= \begin{bmatrix} 0, & -\pi/2, & -\pi/2, & \pi/3, & \pi/3 \end{bmatrix} \\
Wp_5 &= \begin{bmatrix} 0, & 0, & 0, & \pi/2, & \pi/2 \end{bmatrix} \\
Wp_6 &= \begin{bmatrix} 0, & 0, & 0, & -\pi, & \pi \end{bmatrix}
\end{aligned}$$

The number of samples for is computed by the formula below wher T is the total duration(period) of the trajectory and dt the Sampling interval.

$$N_{samples} = T/dt \tag{3.1}$$

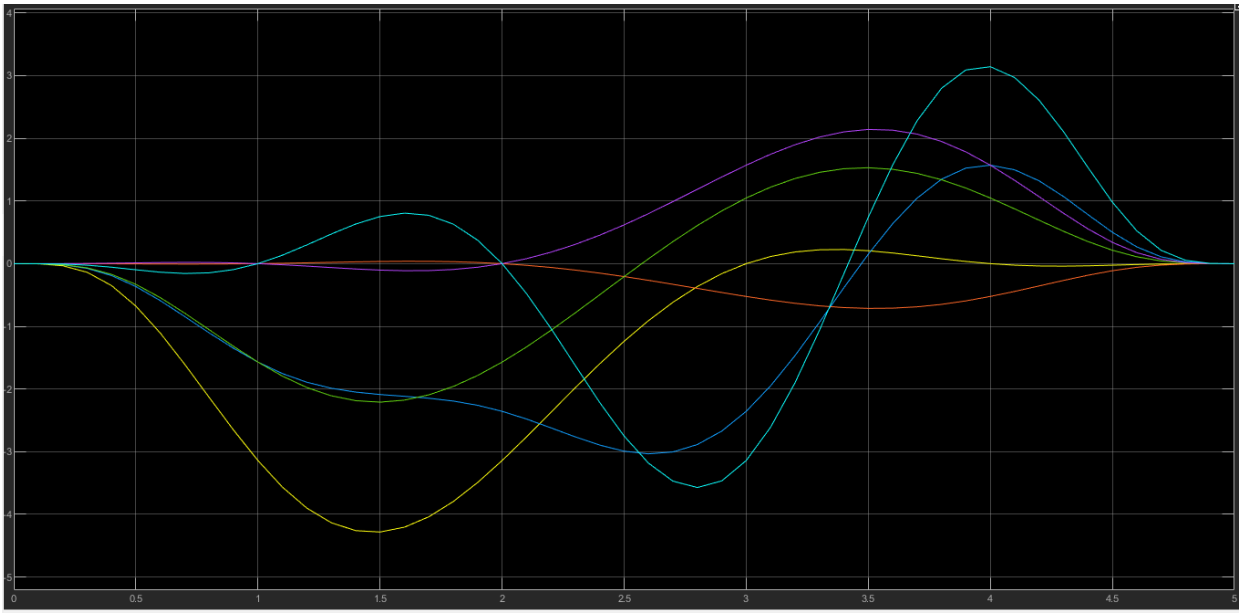


Figure 3.2: Desired Trajectory.

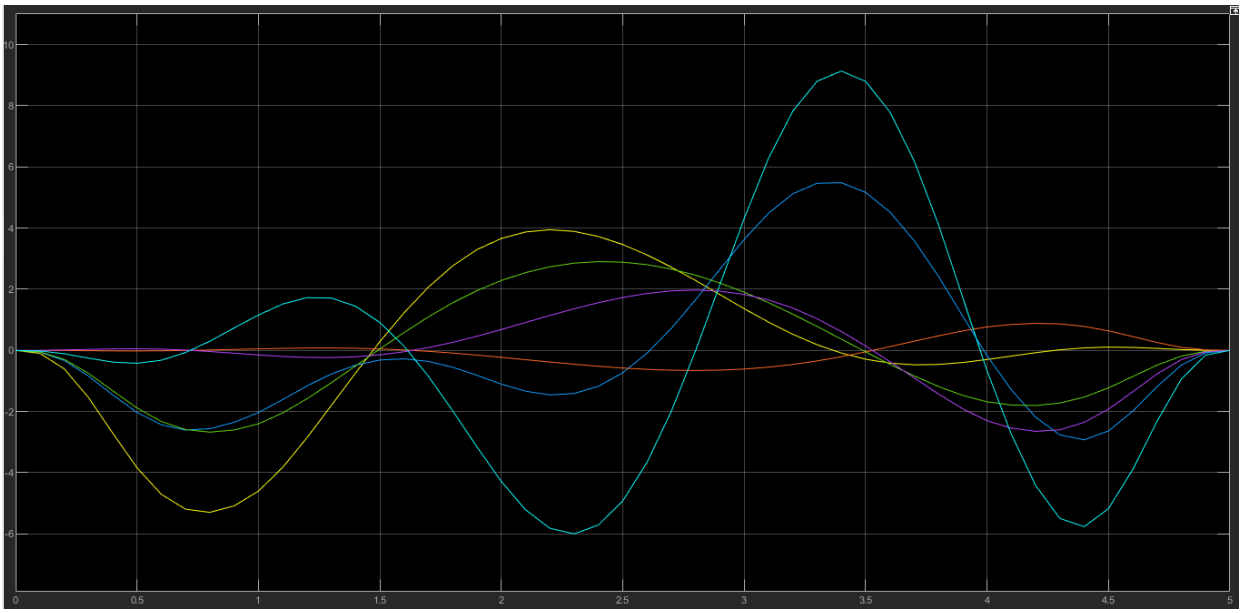


Figure 3.3: Angular Velocity.

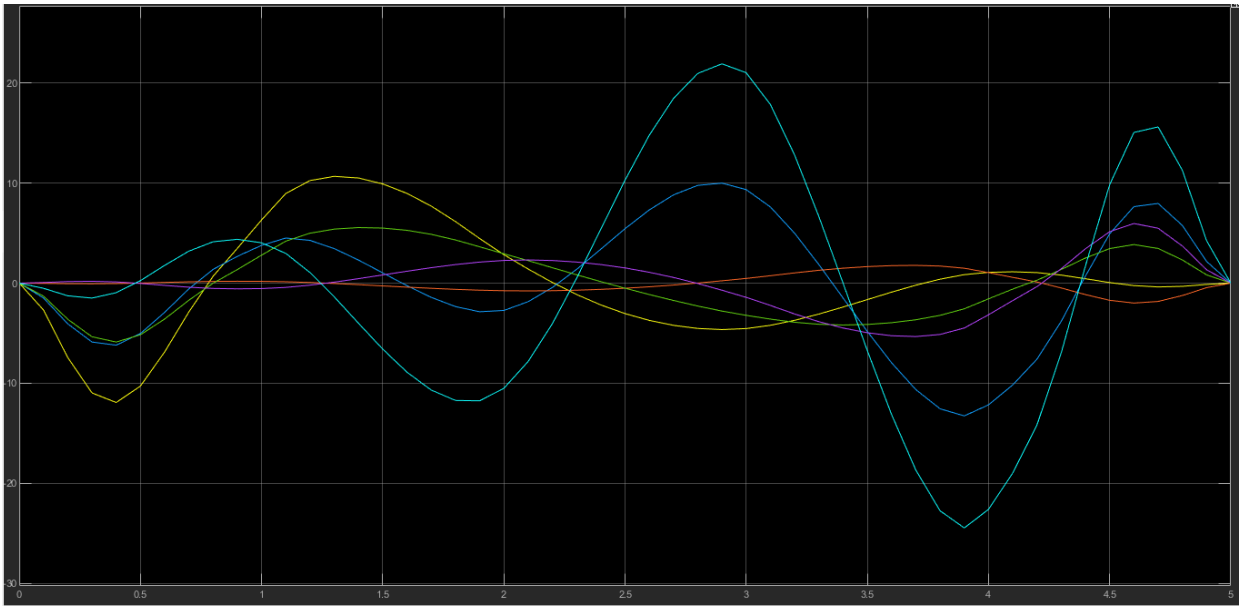


Figure 3.4: Angular acceleration.

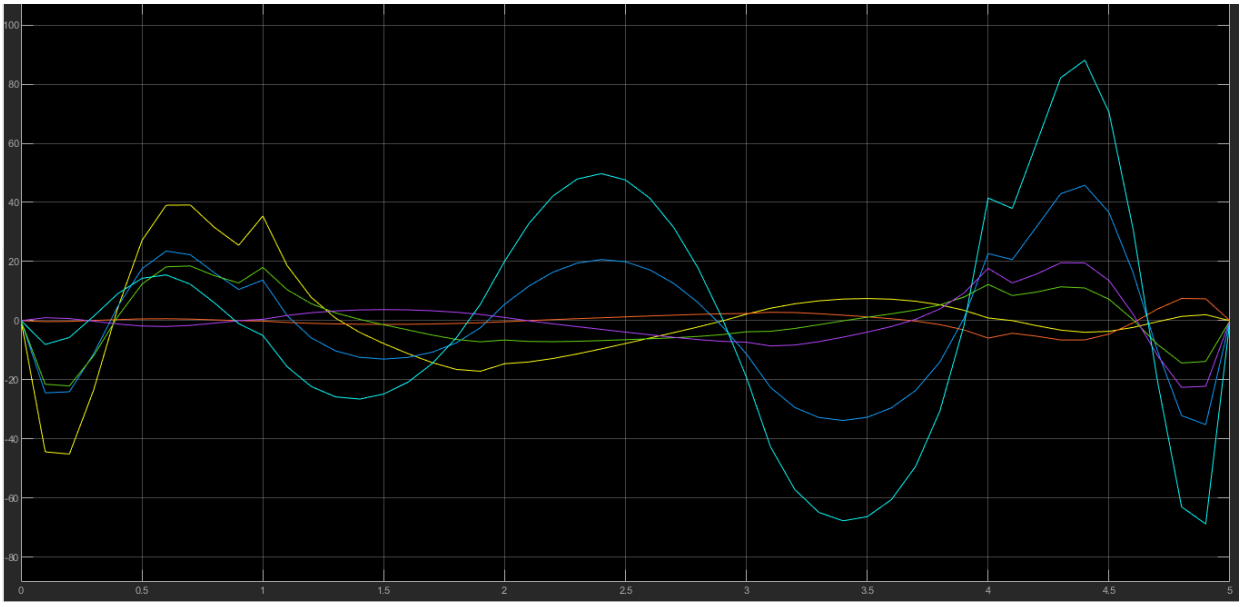


Figure 3.5: Jerks.

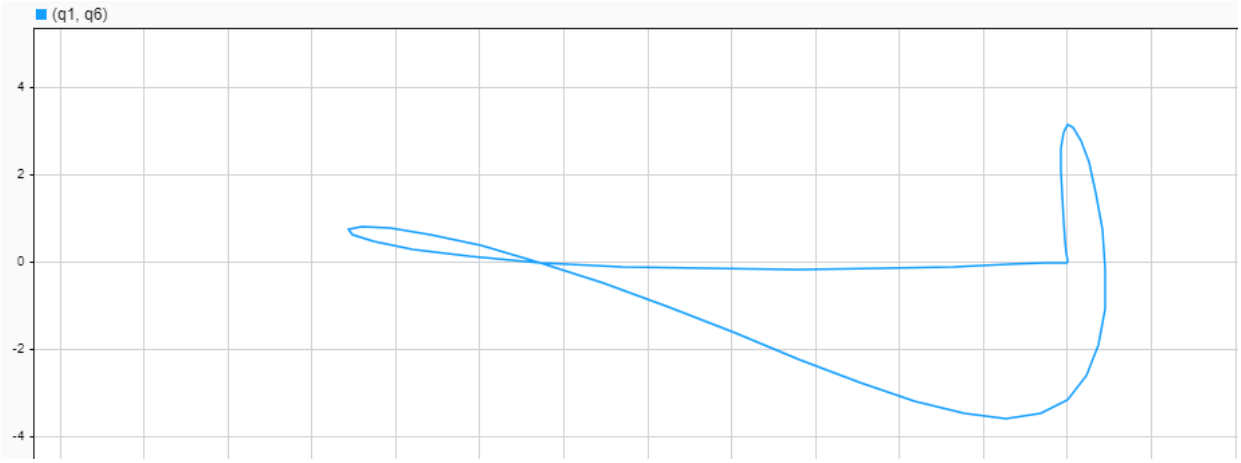


Figure 3.6: q_1 and q_6 positions.

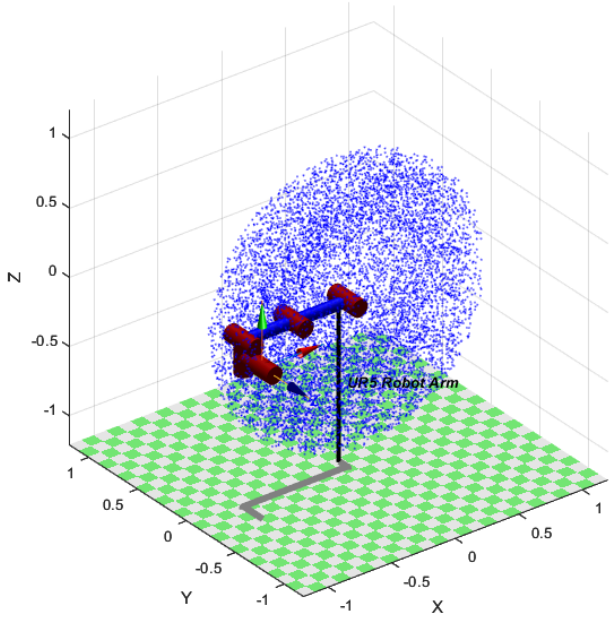


Figure 3.7: UR5 workspace without singularity avoid.

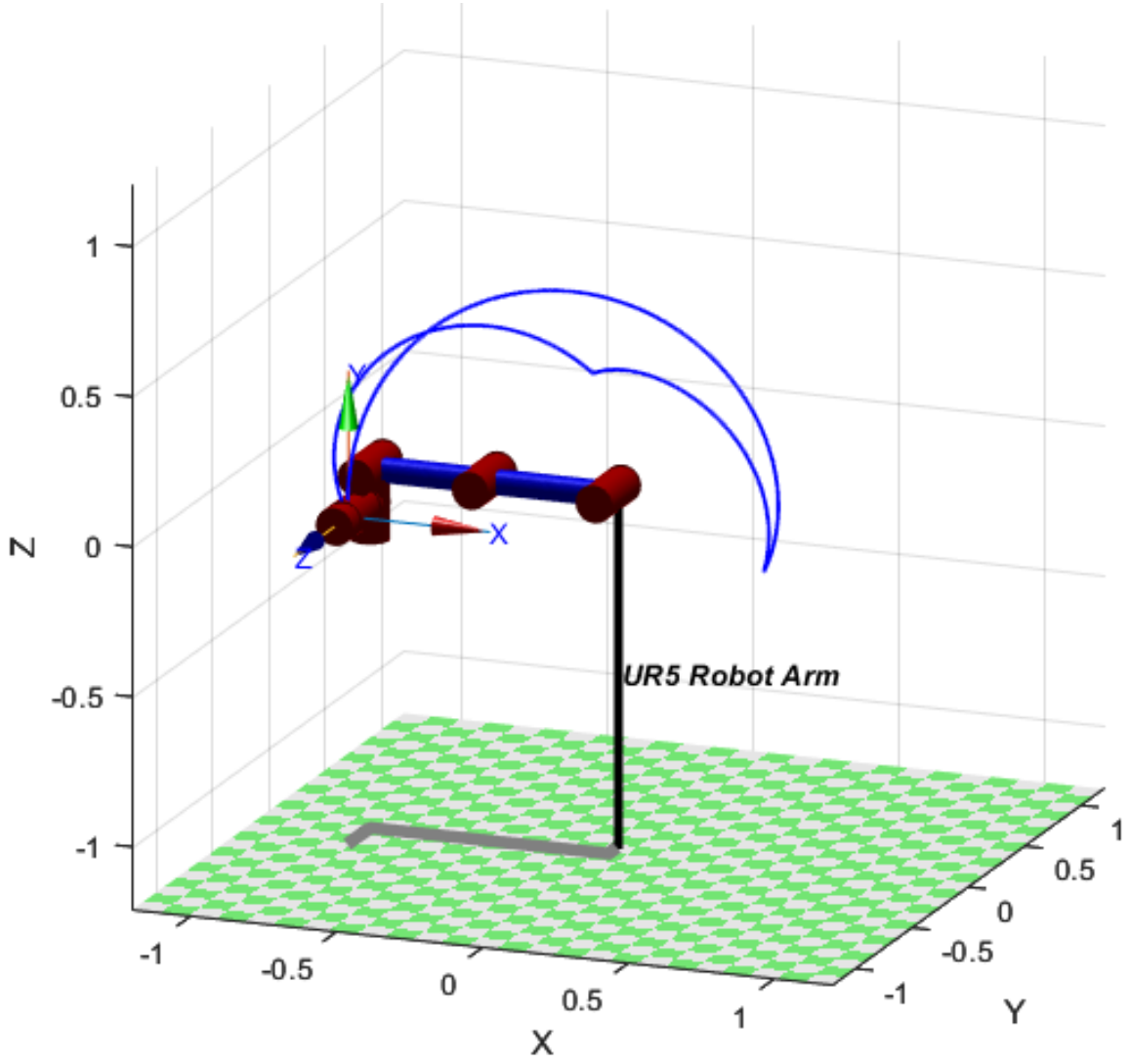
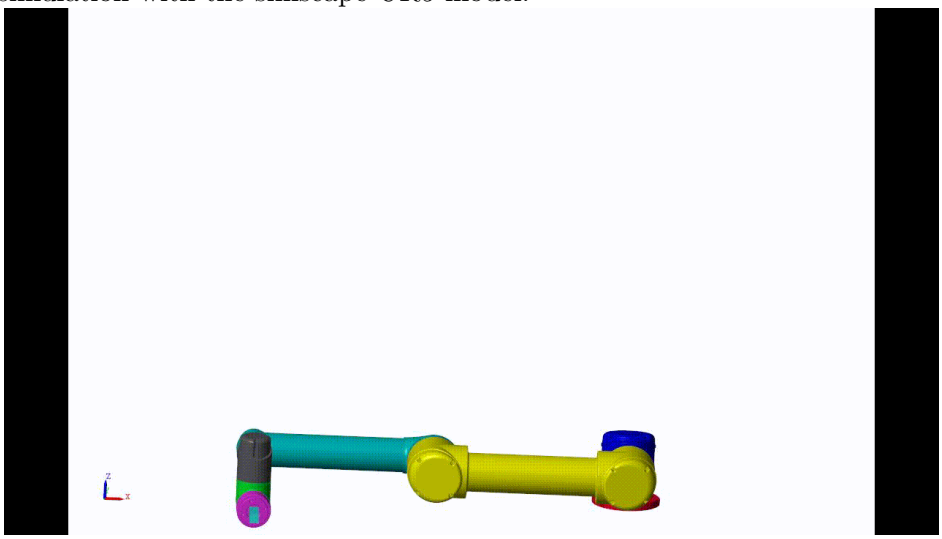


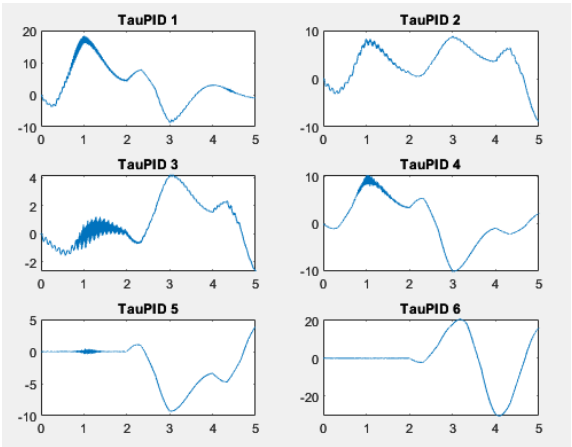
Figure 3.8: End-effector position.

Chapter 4

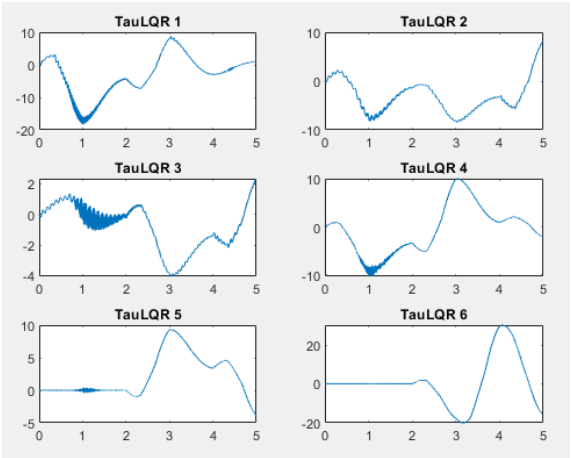
Simulation part

In this section, we are going to present the plots of the result obtained, using the above control solution in simulation with the simscape UR5 model.





(a) Unfiltered torque output from LQR controller.



(b) Unfiltered torque output from PID controller.

Figure 4.1: Delivery torque from each control.

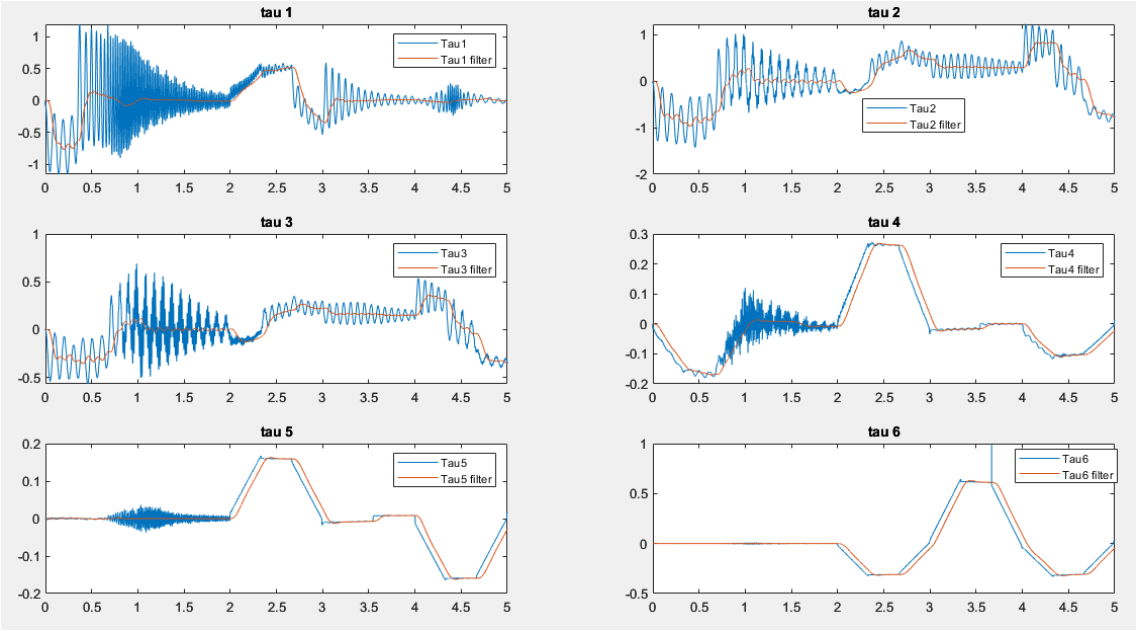
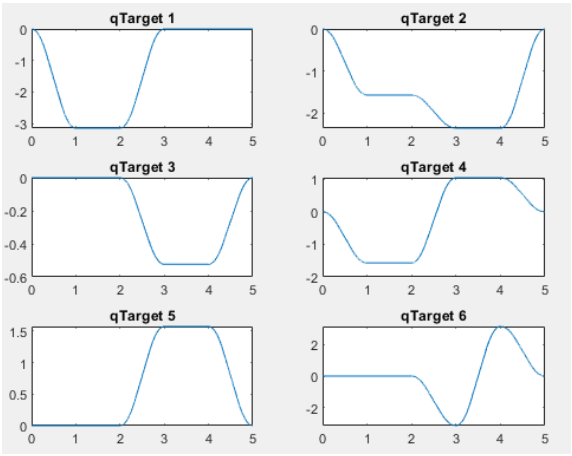
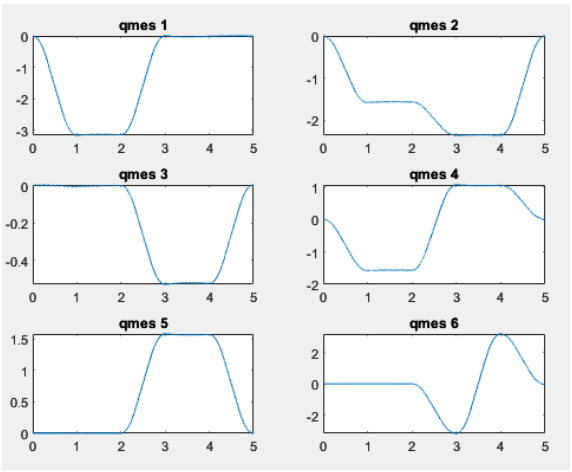


Figure 4.2: Unfilterred Vs filtered torque output from overall controller



(a) path trajectory for each joint.



(b) Trajectory mesured from each joint.

Figure 4.3: Single data response after control design.

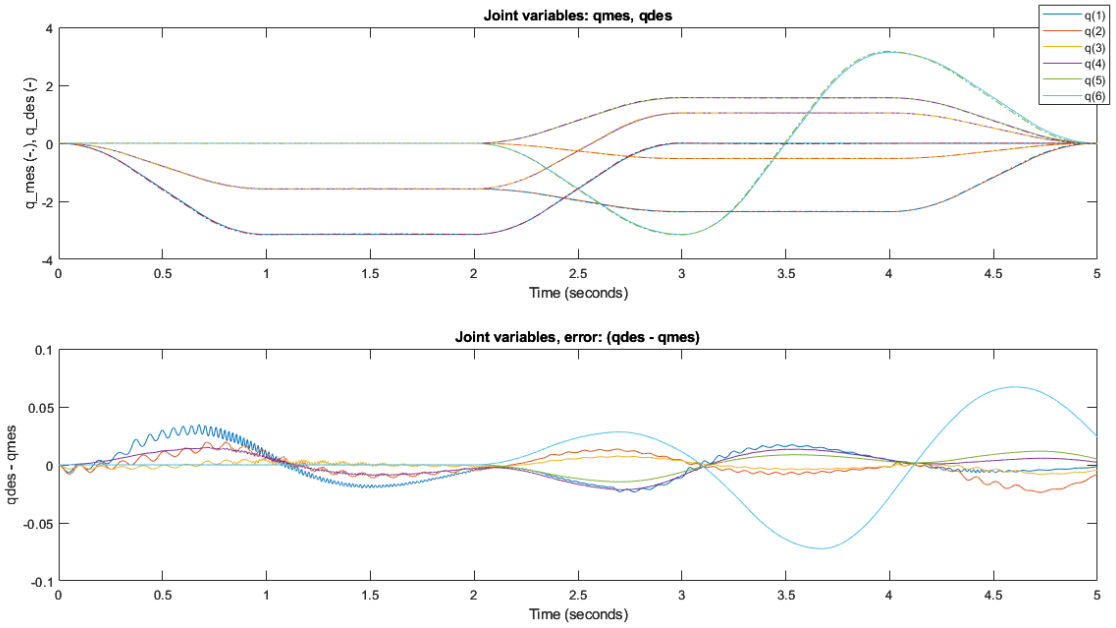


Figure 4.4: Input/ output comparaison and position error.

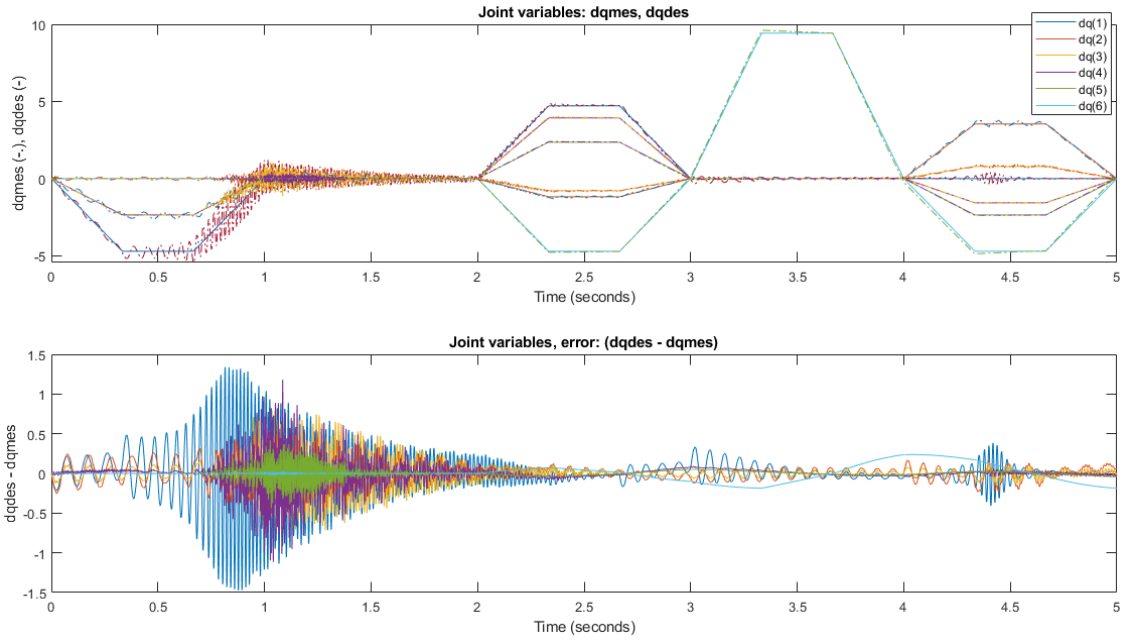


Figure 4.5: Input/ output comparaison and velocity error.

Conclusions

Based on the results of simulations and analysis that have been done in robots UR5 collaboration study using PID and Gravity compensation controller, so there are some conclusion : (1) Dynamic model of collaborative robots UR5 that not controlled yet has the result of the end effector angle position response (serial links) unstable, it's also has steady state error that large enough, as well as angular movement at the end effector it never reaches a steady state. (2) PID controllers + Gravity compensation are applied to the dynamic model to calculate errors between actual positions and setpoints given. Based on the results of the simulation, the PID controller + G Compensation can make the angular position response at the end effector capable achieve a steady state and meet the design criteria given. To ensure the correct behavior also in case an external distubance was been applied, a LQR controller was implemented to tracking the desired goal, minimizing the error and showing the accuracy of the mathematical model.

Bibliography

[1] www.universal-robots.com

[2] M. Sorli, Dynamic parameters identification of a UR5 robot manipulator. Master Thesis in Mechanical Engineering, Politicnico di Torino, 2020.

[3] K. Kufieta, Force Estimation in Robotic Manipulators: Modeling, Simulation and Experiments,UR5 as a case study. Thesis from Norwegian University of Science and Technology, Department of Engineering Cybernetics, Jan 29, 2014.

[4] Slide Modelling and simulation of mechatronic systems of prof.Machelli, AY 2021/22,Unibo.