Modelling and Simulation of Mechatronic Systems M

UR5 Robot Manipulator

Project

Laura Calzoni
Jacopo Merlo Pich
Francesca Paradiso
Agatino Ricciardi

The UR5 robot produced by Universal Robots™ is a six dof anthropomorphic robotic arm designed to work closely with human operators, without the need for physical barriers of separation. This collaborative robot (cobot) is engineered to perform a wide range of tasks in industrial settings, including assembly, packaging, inspection, material handling, and more.

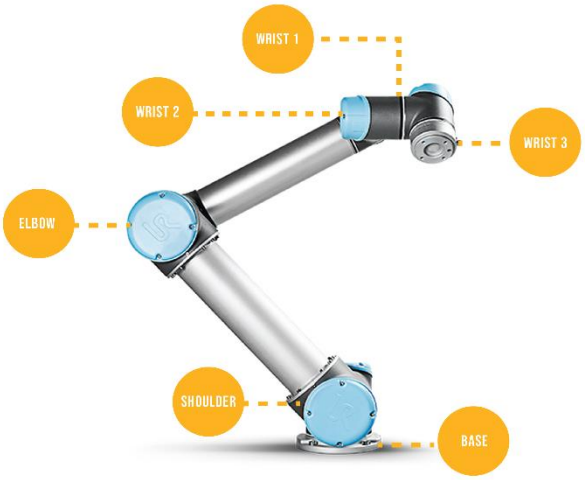


Figure 1 - UR5 robotic arm

It is composed by six links (plus the base) connected by six revolute joints, which can be referred to as Base, Shoulder, Elbow, Wrist 1, Wrist 2 and Wrist 3. The first three are responsible for the positioning of the end effector in the workspace, while the wrist joints control the Tool Center Point (TCP) in the right orientation.
A characteristic of the UR5 is that it does not have a spherical wrist. This solution ensures each joint to have its own motor and, therefore, a transmission system is not necessary.

The aim of this project was to properly model the UR5 robot dynamics in MATLAB Simulink/Simscape Multibody™ environment and control it in order to follow a desired trajectory in the workspace. The developed model includes the reduction gears (one for each joint) while the actuation system was not considered.
All the dynamic and kinematic parameters were obtained from the information provided by the manufacturer, if not specified otherwise.

1) Implementation of the kinematics

The mechanical structure of the UR5 was implemented in Simulink's Simscape Multibody. It is a multibody simulation environment specifically designed for the simulation of 3D mechanical systems and based on the principle of modelling systems as collections of rigid bodies connected through kinematic pairs.

Each link has been approximated to a parallelepiped using the "*Solid*" block. Universal Robots™ provides some parameters regarding inertia, such as the masses of the links and the positions of the centres of mass with respect to the D-H frames:

link	m_i [kg]	$r_{G,i}$ [m]
1	3.7	$(0,\ -0.02561,\ 0.00193)^T$
2	8.393	$(0.2125,\ 0,\ 0.11336)^T$
3	2.33	$(0.15,\ 0,\ 0.0265)^T$
4	1.1219	$(0,\ -0.0018,\ 0.01634)^T$
5	1.1219	$(0,\ 0.0018,\ 0.01634)^T$
6	0.1879	$(0,\ 0,\ -0.001159)^T$

Table 1: Mass and position of the center of gravity of each link

As well as the matrices of inertia of each link:

$$I_1 = \begin{bmatrix} 0.0067 & 0 & 0 \\ 0 & 0.0064 & 0 \\ 0 & 0 & 0.0067 \end{bmatrix}, I_2 = \begin{bmatrix} 0.0149 & 0 & 0 \\ 0 & 0.3564 & 0 \\ 0 & 0 & 0.3553 \end{bmatrix},$$

$$I_3 = \begin{bmatrix} 0.0025 & 0 & 0.0034 \\ 0 & 0.0551 & 0 \\ 0.0034 & 0 & 0.0546 \end{bmatrix}, I_4 = \begin{bmatrix} 0.0012 & 0 & 0 \\ 0 & 0.0012 & 0 \\ 0 & 0 & 0.0012 \end{bmatrix},$$

$$I_5 = \begin{bmatrix} 0.0012 & 0 & 0 \\ 0 & 0.0012 & 0 \\ 0 & 0 & 0.009 \end{bmatrix}, I_6 = \begin{bmatrix} 0.0001 & 0 & 0 \\ 0 & 0.0001 & 0 \\ 0 & 0 & 0.0001 \end{bmatrix}.$$

"Link 0" was arbitrary designed to symbolize a base for the anthropomorphic arm. The mechanical model is composed by six subsystems, each one coming between every pair of links. As it can be seen in Figure 2, each subsystem has four ports: *"Link 0"* and *"Link 1"* (in this example) which represent the physical connection to the previous link and to the following one, *"Joint torque"* that receives the input torque and *"q1, dq1, ddq1"* which provides in output the values of position, velocity, and angular acceleration of the joint.

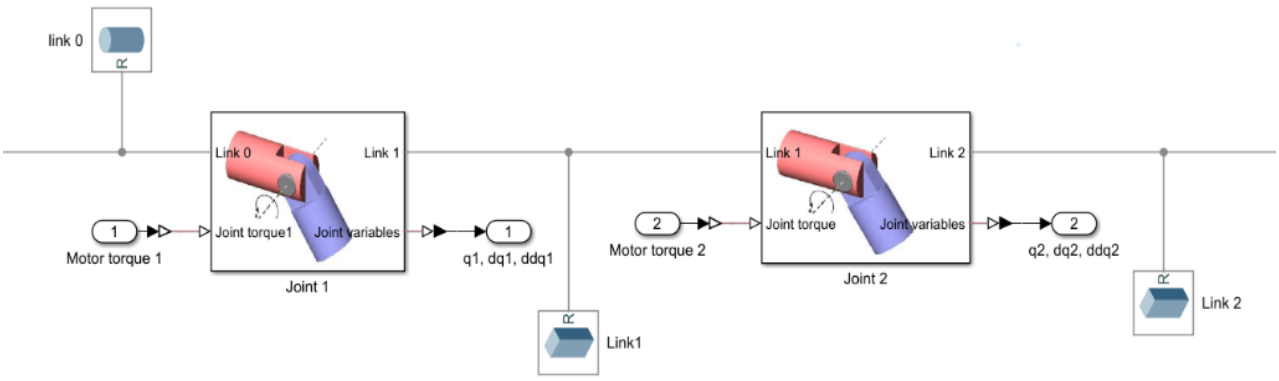


Figure 2 - Kinematic chain of UR5

Each subsystem contains the four rigid transformations defined by the Denavit-Hartenberg (DH) procedure (Figure 3). Multiplied together, they provide the Homogeneous Transformation matrix H_i^{i-1}, that describes the pose of the frame associated to the i-th link with respect to the preceding one.

$$H_i^{i-1} = Trans(z_{i-1}, d_i) \cdot Rot(z_{i-1}, \theta_i) \cdot Trans(x_i, a_i) \cdot Rot(x_i, \alpha_i)$$

Only the second transformation is function of the joint angle, all the others are fixed and therefore realized through the block *"Rigid Transform"*.

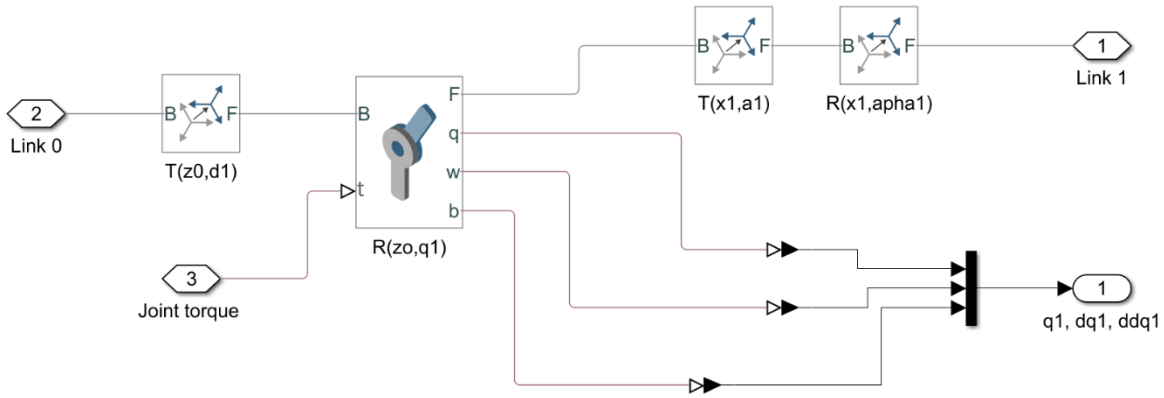


Figure 3 - Foam of subsystem "Joint 1"

The DH parameters provided by Universal Robots™ for the UR5 are reported in Table 2, with q_i $(i = 1, \ldots, 6)$ denoting the degrees of freedom of the robot arm.

Joint	d_i [m]	a_i [m]	α_i [°]	θ_i [°]
Base	0.089159	0	90	q_1
Shoulder	0	-0.425	0	q_2
Elbow	0	-0.39225	0	q_3
Wrist 1	0.10915	0	90	q_4
Wrist 2	0.09465	0	-90	q_5
Wrist 3	0.0823	0	0	q_6

Table 2 – DH parameters of the UR5

To conclude, the robotic arm is then simulated in Simulink.

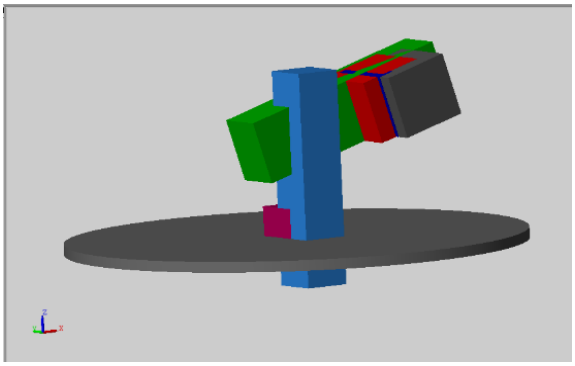


Figure 4 - Simulation of the UR5 in Simulink

2) Harmonic Drive model

As anticipated, in the modelling phase the reduction gear associated to each joint was also considered. In the UR5, the Harmonic Drives are used. The device consists of a thin ring that deflects elastically as it rolls inside a slightly larger rigid circular ring. It has three main elements, concentrically placed around the rotor axis: a circular spline, a flexspline, and a wave generator.

The circular spline is a rigid ring with internal teeth that engage the teeth of the flexspline across the major axis of the wave generator. The flexspline is a thin cylinder with fewer teeth and consequently a smaller effective diameter than the circular spline. The wave generator is ball-bearing assembly elliptical in shape, and acts as a link with two rollers that rotates within the flexspline, causing it to mesh with the circular spline progressively at diametrically opposite points. Each full turn of the wave generator moves the flexspline two teeth in the opposite direction relative to the circular spline, provided that the circular spline is fixed.

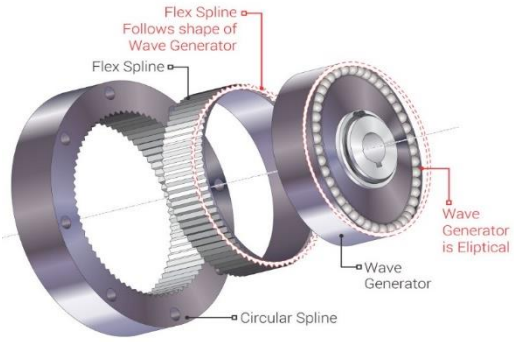


Figure 5 - Harmonic Drive

Among the different types available, our robotic arm features the HFUS-20-2UH on the first three joints, while on the wrist are mounted the HFUS-14-2UH, both chosen with transmission ratio N=100. The first ones have moment of inertia equal to $Jin_{20} = 0.404 * 10^{-4}\ kg \cdot m^2$ while for the second $Jin_{14} = 0.091 * 10^{-4}\ kg \cdot m^2$ (from datasheet).
An approximated mechanical scheme of the harmonic drive could be the following:

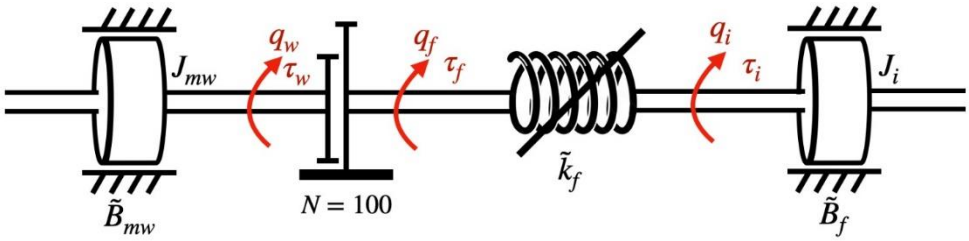


Figure 6 - Harmonic Drive model

With:

- τ_w = torque acting on the wave generator provided by the control action
- q_w = angular position of the wave generator
- J_{mw} = harmonic drive inertia
- $\tilde{B}_{mw}$ = total friction term of the wave generator bearing
- N = gear ratio $(\tau_f = N\,\tau_w)$
- τ_f = torque transmitted to the flexspline
- q_f = flexspline torsion
- $\tilde{k}_f$ = torsional stiffness
- $\tilde{B}_f$ = friction of the flexspline
- q_i = angular position of the i-th joint
- τ_i = torque applied to the joint
- J_i = i-th joint moment of inertia (estimated)

That has been modelled in Simscape as follows:

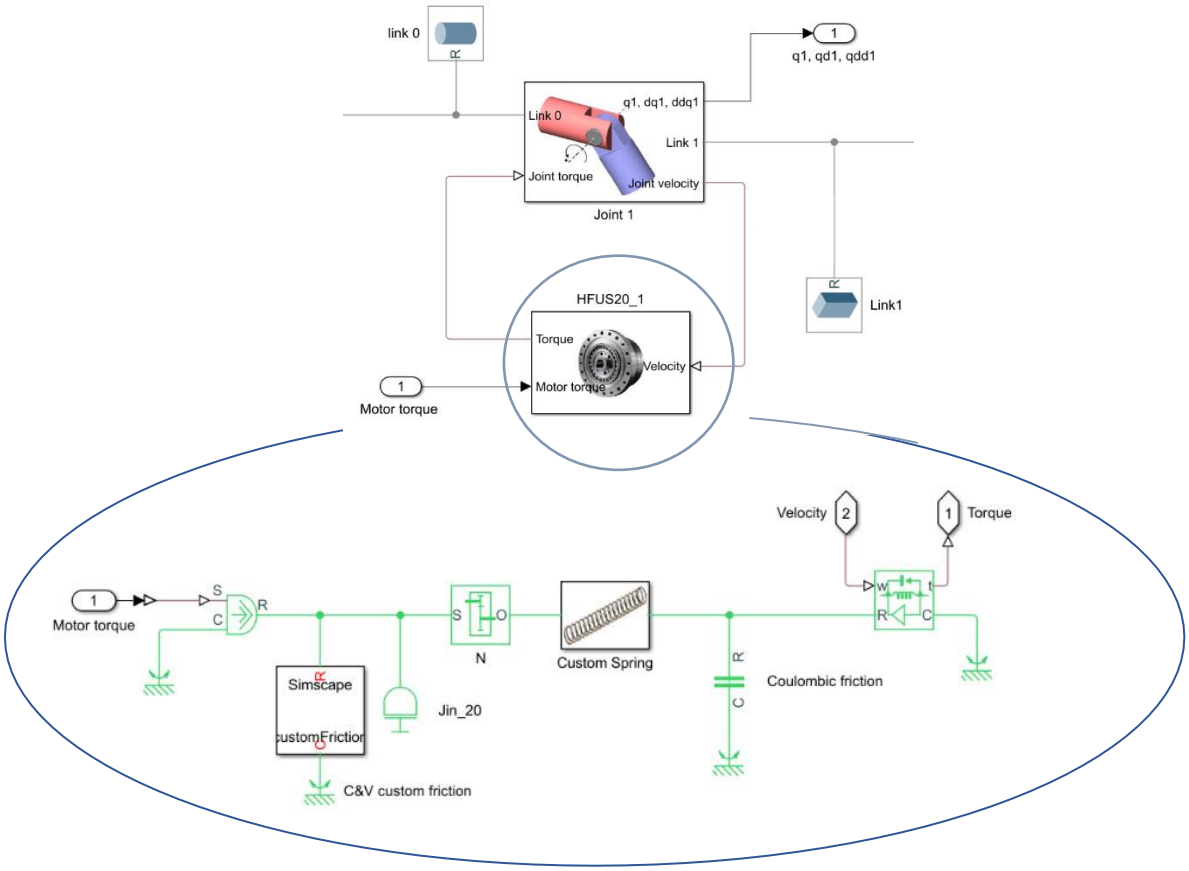


Figure 7 - Simscape realization of the Harmoni Drive (HFUS-20)

The blocks related to the first friction and the stiffness have been custom made (*customSpring20.ssc*, *customSpring14.ssc* and *customFriction.ssc*).

2.1) Torsional stiffness $\widetilde{k}_f(q)$

Experimental observations reveal three physical phenomena related to the flexspline behaviour: nonlinear stiffness, hysteresis, and quasi-backlash due to the soft-windup effect, resulting in the following characteristic:

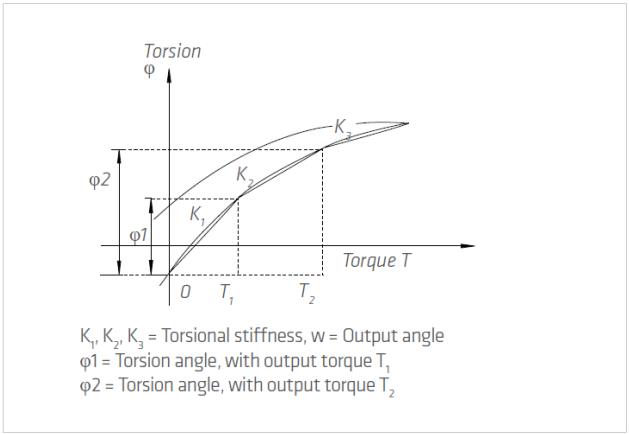


Figure 8 - Torque-torsion curve

The torsional stiffness values *K1*, *K2* and *K3* have been determined by linearizing the curve and averaging the values obtained after numerous tests.

- *K1*: low torque region 0 ~ *T1*
- *K2*: middle torque region *T1* ~ *T2*
- *K3*: high torque region > *T2*

In order to be able to reproduce this characteristic, we had to explicitly calculate the equations of these three lines.

$$\begin{cases} \varphi = \dfrac{T}{K_1} \quad for\ T < T_1 \\ \varphi = \dfrac{T}{K_2} + \widetilde{b_2} \quad for\ T_1 < T < T_2 \\ \varphi = \dfrac{T}{K_3} + \widetilde{b_3}\ for\ T > T_2 \end{cases} \rightarrow \quad \begin{cases} T = K_1 \cdot \varphi \quad for\ \varphi < \varphi_1 \\ T = K_2 \cdot \varphi - b_2 \quad for\ \varphi_1 < \varphi < \varphi_2 \\ T = K_3 \cdot \varphi - b_3 \quad for\ \varphi > \varphi_2 \end{cases}$$

The expressions of the limit torsion angles have been provided by datasheet: $\begin{cases} \varphi_1 = \frac{T_1}{K_1} \\ \varphi_2 = \frac{T_1}{K_1} + \frac{T_2 - T_1}{K_2} \end{cases}$

The offsets instead can be easily calculated as: $\begin{cases} b_2 = K_2\varphi_1 - T_1 \\ b_3 = K_3\varphi_2 - T_2 \end{cases}$

The data have been selected from the datasheet below:

3.3.5 Torsional Stiffness

Table 22.1

	Symbol [Unit]	HFUS-14			HFUS-17			HFUS-20		
Limit torque	T_1 [Nm]		2			3.9			7	
	T_2 [Nm]		6.9			12			25	
Ratio	i [ ]	30	50	≥ 80	30	50	≥ 80	30	50	≥ 80
Torsional Stiffness	K_3 [· 10^3 Nm/rad]	3.4	5.7	7.1	6.7	13	16	11	23	29
	K_2 [· 10^3 Nm/rad]	2.4	4.7	6.1	4.4	11	14	7.1	18	25
	K_1 [· 10^3 Nm/rad]	1.9	3.4	4.7	3.4	8.1	10	5.7	13	16

Figure 9 - Datasheet for the torsional stiffness calculation

2.2) Friction model $\widetilde{B}(\dot{q})$

The model of the friction involves kinetic (Coulomb), viscous, and static friction effects. The static friction includes Dahl and Stribeck terms that can be modelled as exponential or higher-order polynomial functions. The measurement of all these effects is highly complicated. However, since they usually appear only at very low velocities, the following simplified model can be accepted:

$$\tilde{B}_{mw}(\dot{q}_w) = B_{mw}\,\dot{q}_w + (B_{mw}^{+}, B_{mw}^{-})sgn(\dot{q}_w)$$

where B_{mw} is a viscous coefficient, and B_{mw}^{+} and B_{mw}^{-} are positive-direction and negative-direction Coulomb friction torques (equal to B_{mw}^{+} for $\dot{q}_w > 0$, and equal to B_{mw}^{-} for $\dot{q}_w < 0$).
In particular:

- $B_{mw}^{+} = 0.008$ Nm
- $B_{mw}^{-} = -0.007$ Nm
- $B_{mw} = 6.4*10^{-4}$ Nm/(rad/s)

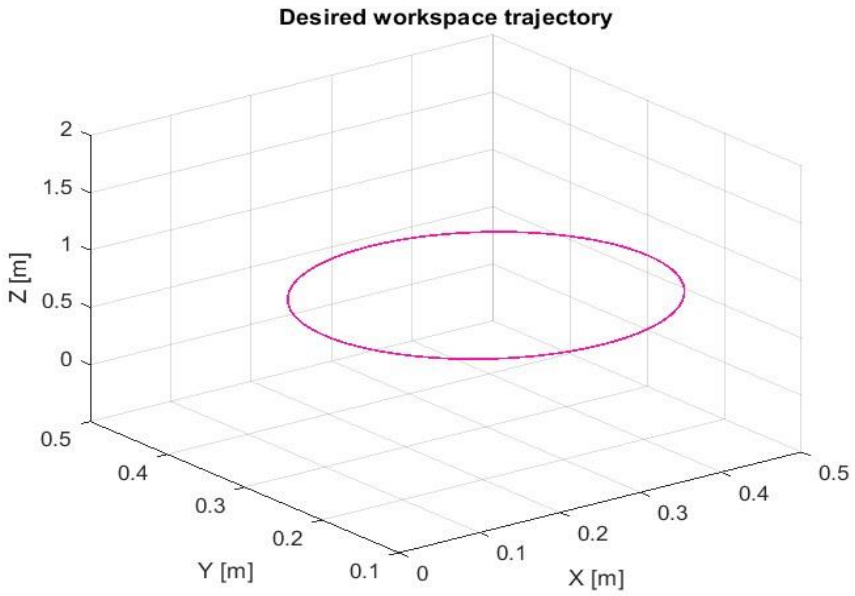
On the other hand, for the flexspline case, if the circular spline is fixed the friction model can be simplified as:

$$\widetilde{B_f} = (B_f^+, B_f^-)sgn(\dot{q}_f)$$

with B_f^+ = 0.4 Nm and B_f^- = -0.4 Nm

3) Inverse kinematics and definition of the desired trajectory

The robot should follow a desired trajectory defined in the workspace and, therefore, the inverse kinematics (IK) of the UR5 must be computed. Indeed, provided the desired motion *desiredPose* defined in the workspace (i.e. in *xyz* coordinates), the inverse kinematics returns the corresponding values $[q_i^{des}, \dot{q}_i^{des}, \ddot{q}_i^{des}]$, with $i = \{1,\ldots,6\}$, for the motion control that takes place in the joint-space. For this purpose, we used the MATLAB function *generalizedInverseKinematics()*. It creates a nonlinear IK solver to calculate the joint configurations using a set of user-defined kinematic constraints based on a specified rigid body tree model. If a solution is possible, also the joint limits specified in the robot model are automatically obeyed. An initial guess for the robot configuration and the desired constraint description objects must be specified. As initial guess we set the home position through *.homeConfiguration*, while as constraint inputs we defined a *constraintPoseTarget* object. This object describes a constraint that requires the pose of one body (the end effector) to match a target pose within a distance and angular tolerance in any direction. In this way, by using the property *.TargetTransform()* it's possible to set *desiredPose* as the pose of the target frame relative to the reference body frame.
As desired motion, instead, we chose to follow an elliptic trajectory in the x-y plane, at 0.65 m of height with a constant angular velocity equal to $\frac{\pi}{1000}$ (in this way the entire path is completed every 2000 samples, i.e. 2s).



ik result.mp4

Figure 10 - Desired workspace trajectory

The following images show the desired jointspace trajectory subsequently obtained from the inverse kinematics.

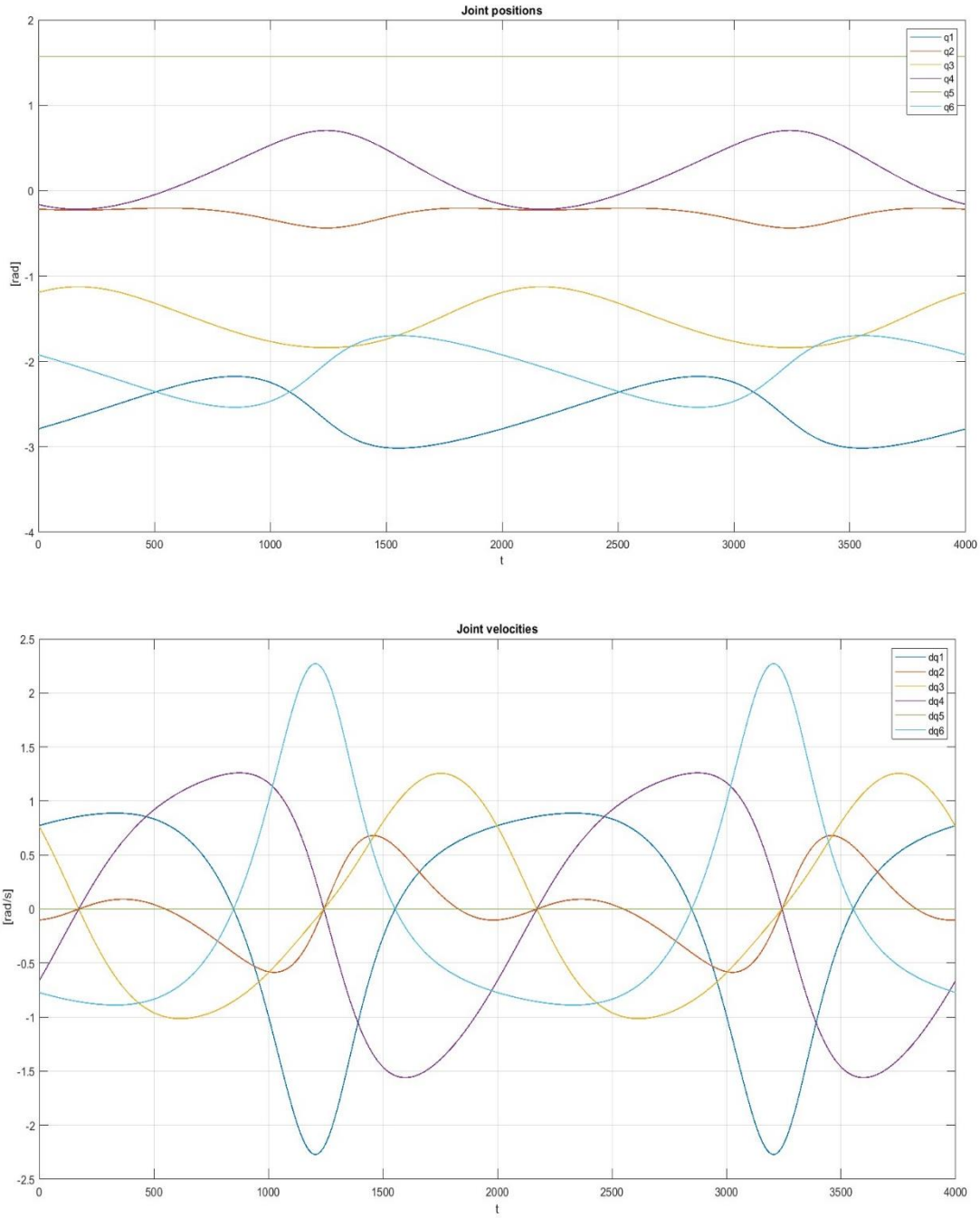


Figure 11 - Desired jointspace trajectories

4) Estimation of the moments of inertia

Before proceeding with the implementation of the motion control, we had to find a way to estimate the moments of inertia J_i felt at each joint, since they are functions of the joints' positions. Indeed, we would underestimate the robot inertia if we took into account just the moments I_i of each link. Considering the Euler-Lagrange model of the UR5:

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + D\dot{q} + G(q) = \tau$$

it can be seen that, for sensing the contribution generated by the inertia forces only to the total torque acting on the manipulator, we had to start moving the joints with zero initial speeding. In this way, it results: $M(q)\ddot{q} + G(q) = \tau$

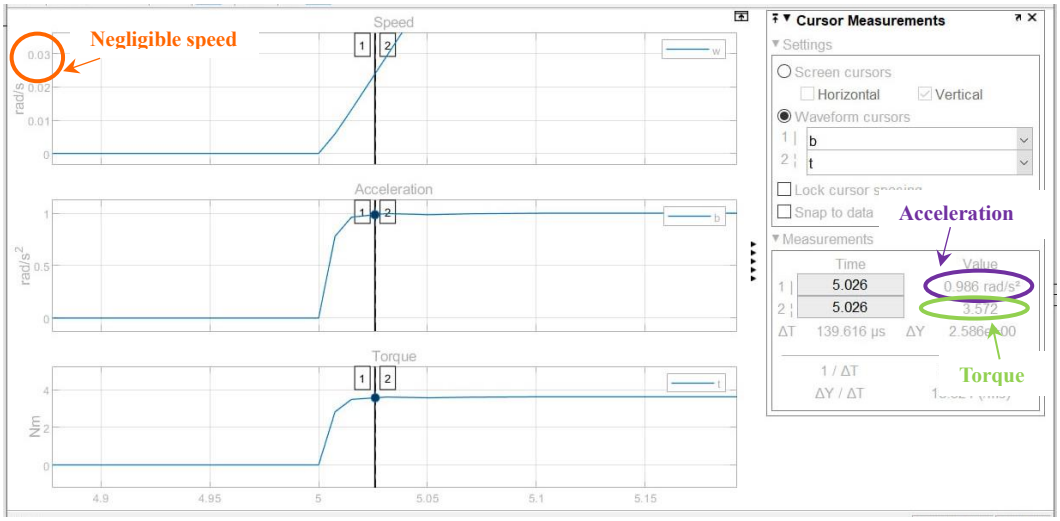


Figure 12 - Measurement of the inertia for Joint 1

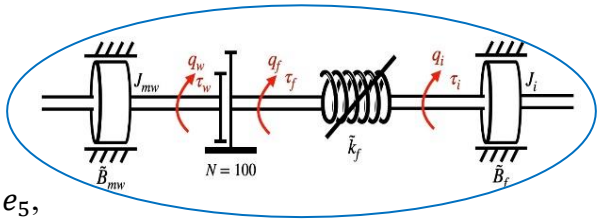
Eliminating the gravity in the simulation environment, it's possible to calculate the elements on the diagonal of $M(q)$ by measuring for each joint the output torque and acceleration, at a time instant for which the velocity can be still neglected. So, we controlled each joint in position with an integrated ramp as reference, and we performed multiple tests, trying different slope values. Then, we have calculated the average of the estimates collected for each joint.

Slopes combination [rad/s²]	M_{11} [kg·m²]	M_{22} [kg·m²]	M_{33} [kg·m²]	M_{44} [kg·m²]	M_{55} [kg·m²]	M_{66} [kg·m²]
[1 0.5 0.75 1 2 1.5]	3,63	5,41	1,43	0,045	$4{,}1 \cdot 10^{-3}$	$2{,}5 \cdot 10^{-4}$
[0.2 1 1 0.5 0.4 0.75]	3,63	3,71	1,82	0,07	0,02	$4{,}3 \cdot 10^{-4}$
[1.5 0.2 0.4 0.1 1 0.5]	3,59	6,04	1,25	0,33	$8{,}1 \cdot 10^{-3}$	$2{,}4 \cdot 10^{-4}$
[0.5 1.5 2 1.75 0.75 1]	3,74	5,1	1,52	0,05	0,02	$6{,}2 \cdot 10^{-4}$
[2 0.4 1.75 0.75 0.2 0.2]	3,64	0,11	0,89	0,11	0,11	$1{,}56 \cdot 10^{-3}$
[0.75 0.75 0.2 2 1.75 0.4]	3,67	3,84	5,37	0,03	$5{,}5 \cdot 10^{-3}$	$8{,}4 \cdot 10^{-4}$
[0.4 2 0.5 0.2 0.5 1.75]	3,72	3,77	5,5	0,23	0,02	$2{,}5 \cdot 10^{-4}$
[1.75 1.75 1.5 1.5 1.5 2]	3,65	4,55	2,04	0,06	$1{,}43 \cdot 10^{-3}$	$3{,}4 \cdot 10^{-4}$
Average	**3,66**	**4,07**	**2,48**	**0,12**	**0,02**	$\mathbf{5{,}66 \cdot 10^{-4}}$

Table 3 - Estimation of the moments of inertia

In particular, it's important to notice that the inertia felt at the last joint remains basically constant over the different experiments. Indeed, since each element $M_{kk}(q)$ is function of the joint positions $q_{k+1}, \ldots, q_N$ only, M_{66} is independent to the current robot configuration.

5) System equations



We have drawn the bond graph of the HD-joint subsystem in order to derive the dynamic equations of the model. For simplicity, the Coulomb friction hasn't been included in the bond graph since, it's value is really low. The characteristic equation of a transformer is $e_4 = n \cdot e_5$, but as the gear ratio equation is $\tau_f = N \cdot \tau_w$, it results $n = \frac{1}{N}$. From now on, we will assume just small deformations of the torsional spring, so just the first part of the HD torsional stiffness characteristics must be considered, i.e. $k_f \equiv K_1$.

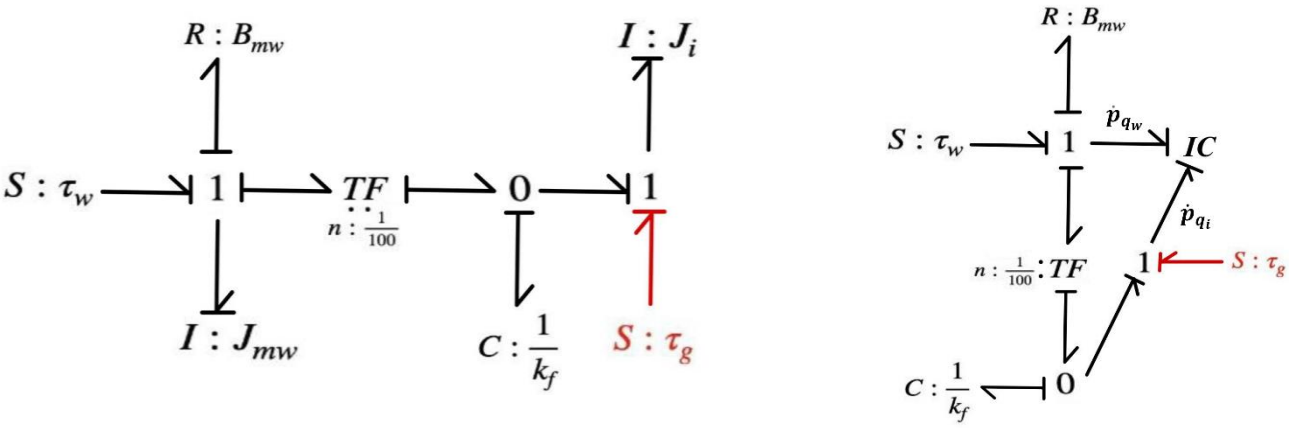


Figure 13 - Bond graph of the model

It's important to consider in the model also the torque due to the gravity force τ_g acting on each joint. We used the Lagrangian approach. The kinetic coenergy of the system can be calculated as:

$$T = \frac{1}{2} J_{mw} \dot{q}_w{}^2 + \frac{1}{2} J_i \dot{q}_i{}^2$$

while the potential energy is:

$$U = \frac{1}{2} k_f s^2 = \frac{1}{2} k_f \left(\frac{q_w}{N} - q_i \right)^2$$

being, $s = q_f - q_i = \frac{q_w}{N} - q_i$

So, it follows:

$$p_{q_w} = \frac{\partial T}{\partial \dot{q}_w} = J_{mw} \dot{q}_w \quad \rightarrow \quad \dot{p}_{q_w} = J_{mw} \ddot{q}_w$$

$$p_{q_i} = \frac{\partial T}{\partial \dot{q}_i} = J_i \dot{q}_i \quad \rightarrow \quad \dot{p}_{q_i} = J_i \ddot{q}_i$$

Given the generalized forces $\begin{cases} E_{q_w} = \tau_w - B_{mw} \dot{q}_w - \frac{\partial U}{\partial q_w} = \tau_w - B_{mw} \dot{q}_w - \frac{k_f}{N} \left(\frac{q_w}{N} - q_i \right) \\ E_{q_i} = \tau_g - \frac{\partial U}{\partial q_i} = \tau_g - k_f \left(\frac{q_w}{N} - q_i \right) \end{cases}$ that include both external and potential forces, then it results:

$$\dot{p}_{q_w} = E_{q_w} + \frac{\partial T}{\partial q_w} = \tau_w - B_{mw}\dot{q}_w - \frac{k_f}{N}\left(\frac{q_w}{N} - q_i\right) = J_{mw}\ddot{q}_w$$
$$\dot{p}_{q_i} = E_{q_i} + \frac{\partial T}{\partial q_i} = \tau_g + k_f\left(\frac{q_w}{N} - q_i\right) = J_i\ddot{q}_i$$

Which leads to: $\begin{cases} \ddot{q}_w = \frac{\tau_w}{J_{mw}} - \frac{B_{mw}\dot{q}_w}{J_{mw}} + \frac{k_f}{NJ_{mw}}q_i - \frac{k_f}{N^2J_{mw}}q_w \\ \ddot{q}_i = -\frac{k_f}{J_i}q_i + \frac{k_f}{NJ_i}q_w + \frac{\tau_g}{J_i} \end{cases}$

Given the state vector $x = [q_w,\ \dot{q}_w,\ q_i,\ \dot{q}_i]$, and the input $u = \tau_w$, system dynamics can be rewritten as a linear state space model in matrix form $\dot{x} = Ax + Bu$:

$$\begin{bmatrix} \dot{q}_w \\ \ddot{q}_w \\ \dot{q}_i \\ \ddot{q}_i \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{k_f}{N^2J_{mw}} & -\frac{B_{mw}}{J_{mw}} & \frac{k_f}{NJ_{mw}} & 0 \\ 0 & 0 & 0 & 1 \\ \frac{k_f}{NJ_i} & 0 & -\frac{k_f}{J_i} & 0 \end{bmatrix} \begin{bmatrix} q_w \\ \dot{q}_w \\ q_i \\ \dot{q}_i \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ \frac{1}{J_{mw}} & 0 \\ 0 & 0 \\ 0 & \frac{1}{J_i} \end{bmatrix} \begin{bmatrix} \tau_w \\ \tau_g \end{bmatrix}$$

With:

$\dot{x} = [\dot{q}_w,\ \ddot{q}_w,\ \dot{q}_i,\ \ddot{q}_i]$

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{k_f}{N^2J_{mw}} & -\frac{B_{mw}}{J_{mw}} & \frac{k_f}{NJ_{mw}} & 0 \\ 0 & 0 & 0 & 1 \\ \frac{k_f}{NJ_i} & 0 & -\frac{k_f}{J_i} & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 0 & 0 \\ \frac{1}{J_{mw}} & 0 \\ 0 & 0 \\ 0 & \frac{1}{J_i} \end{bmatrix}$$

6) Decentralized position control

As already said, our goal is to follow a desired trajectory defined through the inverse kinematics and represented by $x^{des} = \left[{q_w}^{des},\ {\dot{q}_w}^{des},\ {q_i}^{des},\ {\dot{q}_i}^{des}\right]$. It's important to notice that from the inverse dynamics we explicitly know just ${q_i}^{des}$, ${\dot{q}_i}^{des}$ (and $\ddot{q}_i^{des}$). For simplicity, we implemented a decentralized control scheme, where each joint (with the corresponding Harmonic Drive) is independently controlled like a SISO system. The drawback of this approach is that it does not take into account all the coupling effects and the non-linearities that characterize the dynamic model of the UR5. However, it's possible to compute offline a feedforward action able to partially compensate for such effects.

The error vector is defined as $e = x - x^{des}$. Consequently:

$$\dot{e} = \dot{x} - \dot{x}^{des} = Ax + Bu - \dot{x}^{des} = A(e + x^{des}) + Bu - \dot{x}^{des} = Ae + Bu + Ax^{des} - \dot{x}^{des}$$

The input u shall be composed by both a feedforward term u_{ff} to improve the performances of the control action and a feedback action u_{fb} to cancel the error:

$$u = u_{ff} + u_{fb} = u_{ff} + Ke$$

Thus,

$$\dot{e} = Ae + B(u_{fb} + u_{ff}) + Ax^{des} - \dot{x}^{des} = (Ae + Bu_{fb}) + (Bu_{ff} + Ax^{des} - \dot{x}^{des})$$

$$\boxed{\dot{e} = (A + BK)e + (Bu_{ff} + Ax^{des} - \dot{x}^{des})}$$

From the latter formula we can extract information about how to properly design the control input:

1. the feedforward action u_{ff} must be implemented such that $Bu_{ff} + Ax^{des} - \dot{x}^{des} = 0$
2. the gain matrix K must be designed such that $(A + BK)$ Hurwitz;

In this way, indeed, the error dynamics will asymptotically converge to zero.

6.1) Feedforward action

The feedforward action must be designed in order to cancel out the residual dynamics of the desired trajectory: $Bu_{ff} + Ax^{des} - \dot{x}^{des} = 0$

Substituting,

$$\begin{bmatrix} 0 & 0 \\ \frac{1}{J_{mw}} & 0 \\ 0 & 0 \\ 0 & \frac{1}{J_i} \end{bmatrix} \cdot \begin{bmatrix} u_{ff} \\ \tau_g \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{k_f}{N^2 J_{mw}} & -\frac{B_{mw}}{J_{mw}} & \frac{k_f}{NJ_{mw}} & 0 \\ 0 & 0 & 0 & 1 \\ \frac{k_f}{NJ_i} & 0 & -\frac{k_f}{J_i} & 0 \end{bmatrix} \begin{bmatrix} q_w^{des} \\ \dot{q}_w^{des} \\ q_i^{des} \\ \dot{q}_i^{des} \end{bmatrix} - \begin{bmatrix} \dot{q}_w^{des} \\ \ddot{q}_w^{des} \\ \dot{q}_i^{des} \\ \ddot{q}_i^{des} \end{bmatrix} = 0$$

I get the following 4 equations:

$$\begin{cases} \dot{q}_w^{des} - \dot{q}_w^{des} = 0 \\ \frac{u_{ff}}{J_{mw}} - \frac{k_f}{N^2 J_{mw}} q_w^{des} - \frac{B_{mw}}{J_{mw}} \dot{q}_w^{des} + \frac{k_f}{NJ_{mw}} q_i^{des} - \ddot{q}_w^{des} = 0 \\ \dot{q}_i^{des} - \dot{q}_i^{des} = 0 \\ \frac{k_f}{NJ_i} q_w^{des} - \frac{k_f}{J_i} q_i^{des} - \ddot{q}_i^{des} + \frac{\tau_g}{J_i} = 0 \end{cases}$$

$$\begin{cases} u_{ff} = \frac{k_f}{N^2} q_w^{des} + B_{mw} \dot{q}_w^{des} - \frac{k_f}{N} q_i^{des} + J_{mw} \ddot{q}_w^{des} \\ q_w^{des} = N q_i^{des} + \frac{NJ_i}{k_f} \ddot{q}_i^{des} - \frac{N}{k_f} \tau_g \end{cases}$$

Therefore, the feedforward term needed to be implemented as:

$$\boxed{u_{ff} = \frac{k_f}{N^2} q_w^{des} + B_{mw} \dot{q}_w^{des} - \frac{k_f}{N} q_i^{des} + J_{mw} \ddot{q}_w^{des}}$$

With the desired trajectory of the wave generator given by:

$$q_w^{des} = Nq_i^{des} + \frac{NJ_i}{k_f}\ddot{q}_i^{des} - \frac{N}{k_f}\tau_g$$

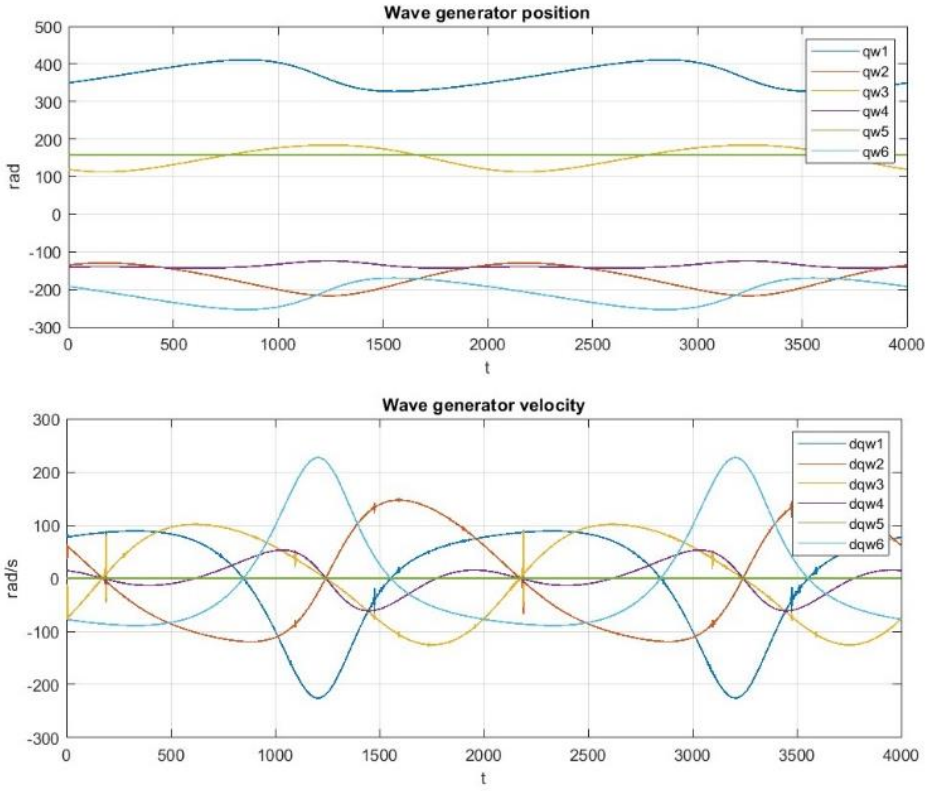


Figure 14 - Reference trajectories for the WG

It must be noticed that inside the feedforward action is present the gravity term that is unknown, up to now. As a consequence, it must be estimated in order to be able to execute the compensation.

6.2) Compensation of the gravity term

As mentioned above, the feedforward action includes the compensation for the gravity forces acting on the manipulator. As the torque generated on each joint by the gravity force depends on the actual robot's configuration, we had to estimate it in an experimental way (as previously done for the inertia matrix).
From the Euler-Lagrange model of the UR5:

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + D\dot{q} + G(q) = \tau$$

it can be seen that for sensing the torque developed by the gravitational action, we need to fix the robot in a standstill configuration $q^{ref} = \begin{bmatrix} q_1^{ref} & q_2^{ref} & q_3^{ref} & q_4^{ref} & q_5^{ref} & q_6^{ref} \end{bmatrix}^T$, since, in this way, $\dot{q} = \ddot{q} = 0$ and the dynamics reduces to:

$$G(q^{ref}) = \tau$$

Therefore, by controlling the joints in position and injecting the reference jointspace trajectories, the output torque sensed at each joint would easily provide the profile of the corresponding gravity torque, developed during the motion.

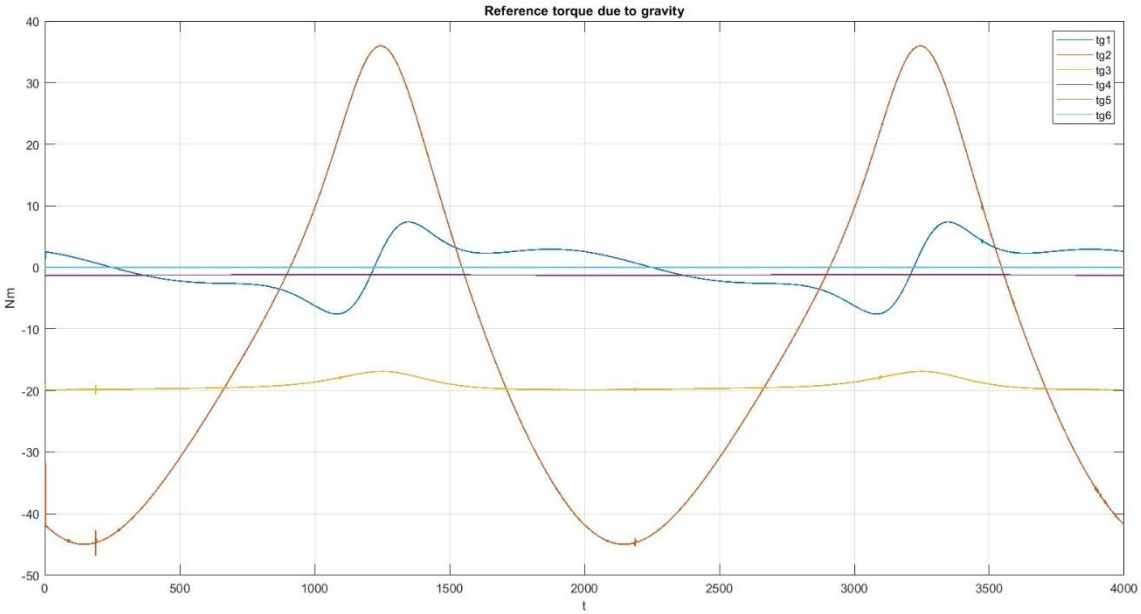


Figure 15 - Gravity torque developed for the reference motion

The torque generated by the gravity force on the sixth joint is always zero probably because the center of mass of the last link is practically coincident with the center of the joint and both its mass and inertia matrix are very small.

6.3) Feedback action

The feedback action is proportional to the position error $u_{fb} = Ke$ and, as said, the gain matrix K must be designed such that the system $(A + BK)$ is Hurwitz. It must be underlined that in the system the gravity term must not be anymore included, since its contribution has already been compensated by the feedforward action. For the purpose, we relied on the *lqr()* MATLAB function. Provided the state-space matrices *A* and *B* of a continuous-time system, it calculates the optimal gain matrix *K*, the solution *S* of the related algebraic Riccati equation and the closed-loop poles *P* associated. The weight matrices for states and inputs (*Q* and *R*, respectively) must be specified too. Larger weights in the *Q* matrix indicate that the corresponding state variables are more important and should not deviate from their desired behavior. Instead, in *R* larger values discourage larger control inputs and excessive control effort.

As a consequence, we have assigned smaller weights to the state variables $\dot{q}_w{}^{des}$ and $\dot{q}_i{}^{des}$ since we are more concerned about tracking just the desired position. Similarly, we have given higher cost to the inputs that control the first three joints, as we wish the system to react quickly in terms of positioning in the desired way, with respect to assure the desired orientation.

Overall, the total control scheme has been implemented in Simulink as displayed below:

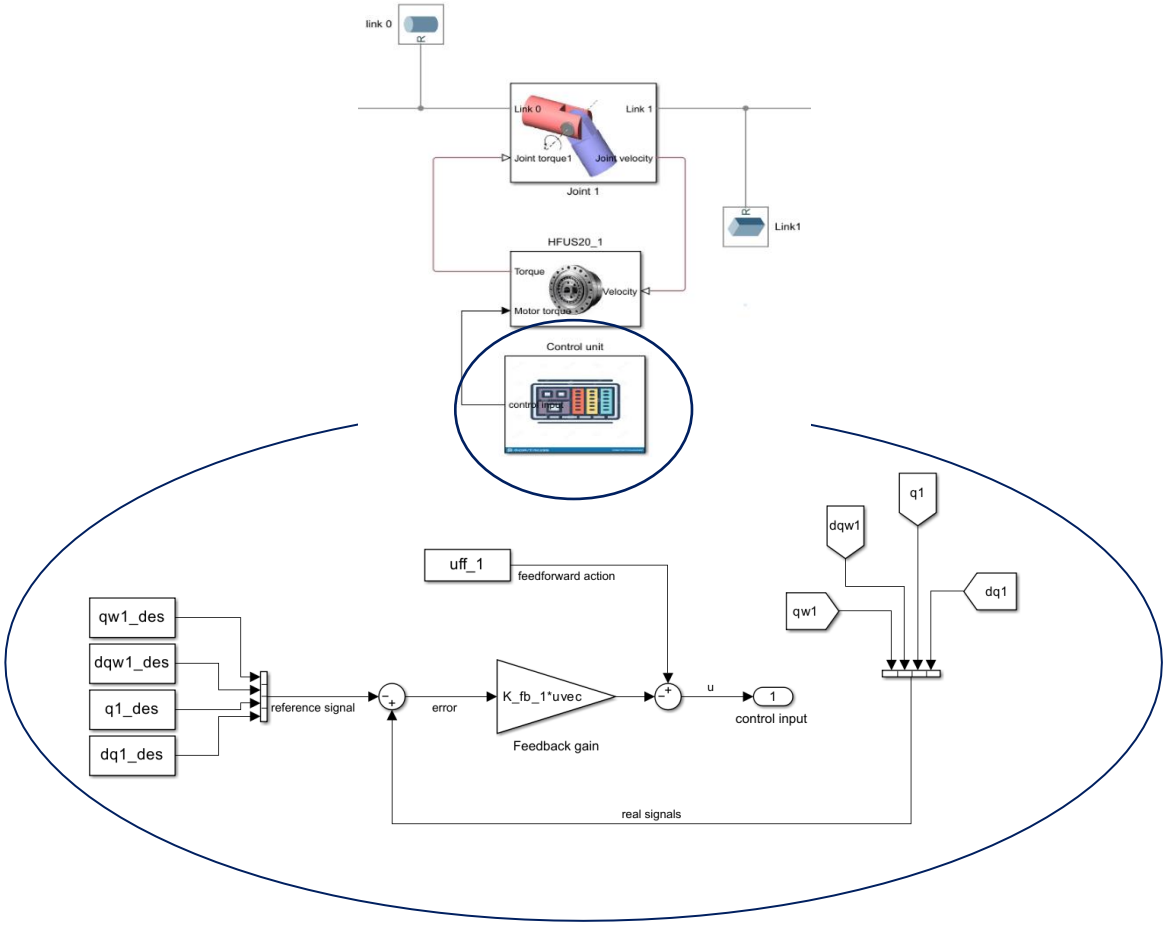
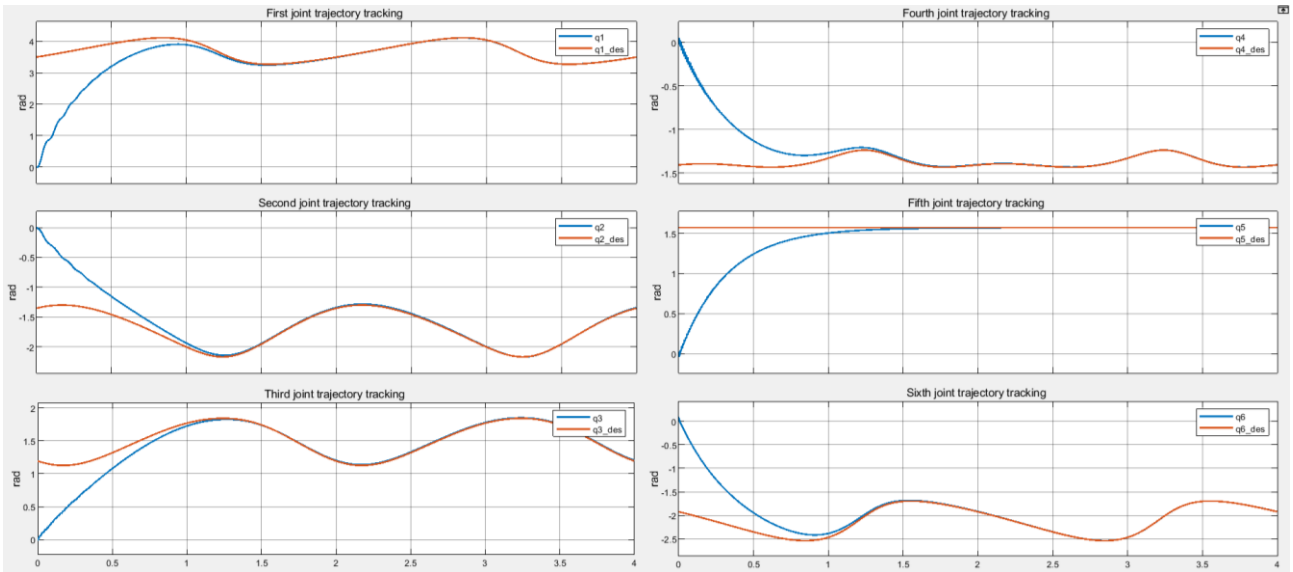


Figure 16 - Control unit of Joint 1

7) Results

With the control strategy just presented we have obtained great results in the joint-space as well as in the workspace:

C:\Users\laura\OneDrive\Desktop\UR5\UR5_simulink.avi



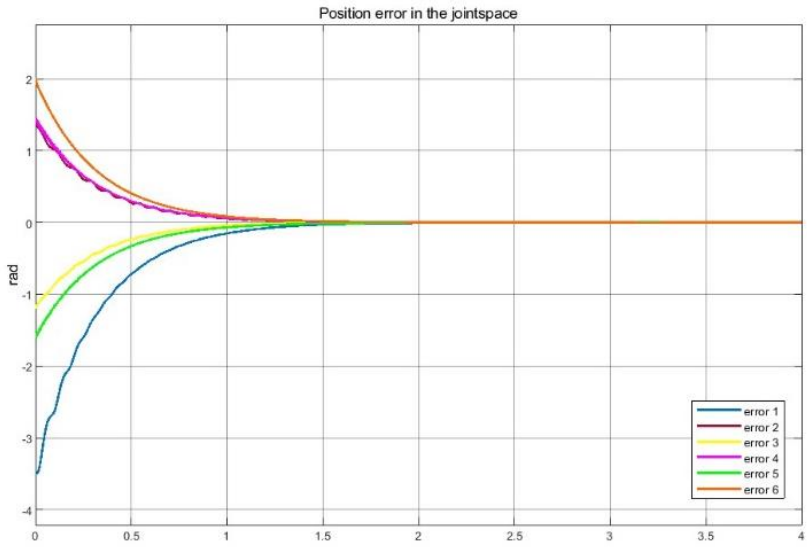


Figure 17 - Performances in the jointspace

To calculate the error in the workspace, we had to get the joints' trajectories from Simulink and remap the motion to the *xyz-p*lane using the forward kinematics. In particular, we have the exploited the function *.getTransform()* to compute the transformation from the end-effector frame to the base frame, for each configuration the robot assumes during the motion.

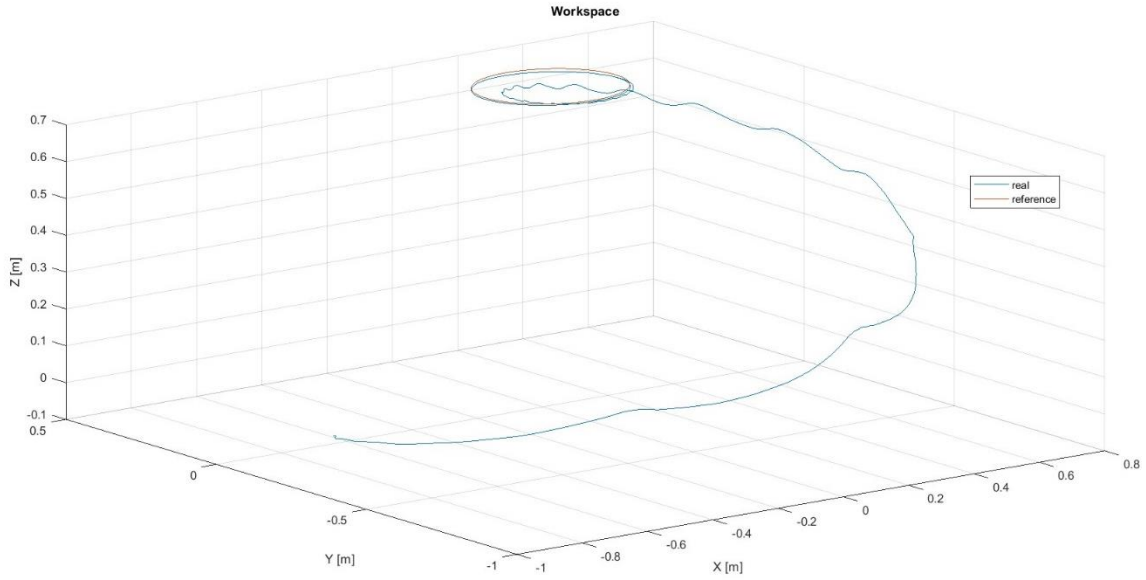


Figure 18 - Motion in xyz-coordinates

As we can see from the picture below, the “steady state” errors in the workspace are of few millimetres.

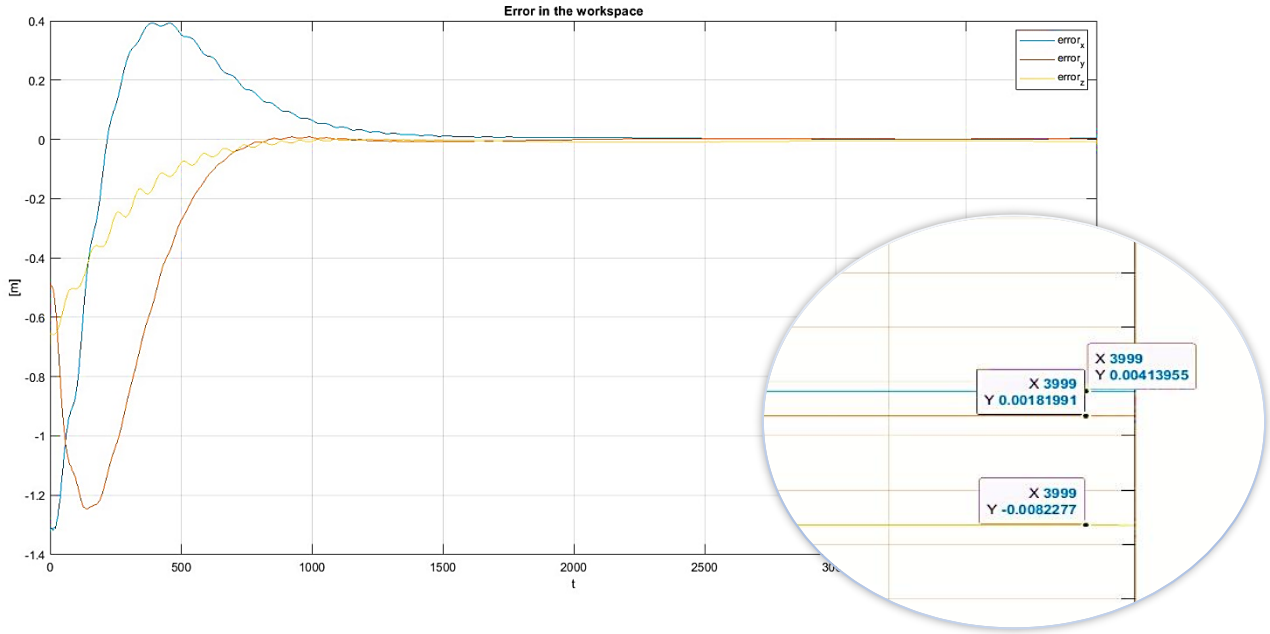


Figure 19 - Performances in the workspace