Robotics Project

Kinematic Control of Robotic Arm

Nikolaos Konstas

Department of Electrical & Computer Engineering
Aristotle University of Thessaloniki

June 2023

Contents

1 Part A

1.1 Introduction

In this work we deal with the kinematic control of a 6-degree-of-freedom arm. More specifically, we have a ur10e arm, which accepts speed commands, and has a maximum permissible absolute joint speed of $\begin{bmatrix}120 & 120 & 180 & 180 & 180 & 180\end{bmatrix}^\circ /sec$, and a maximum acceleration of 250 rad/s^2 per joint.

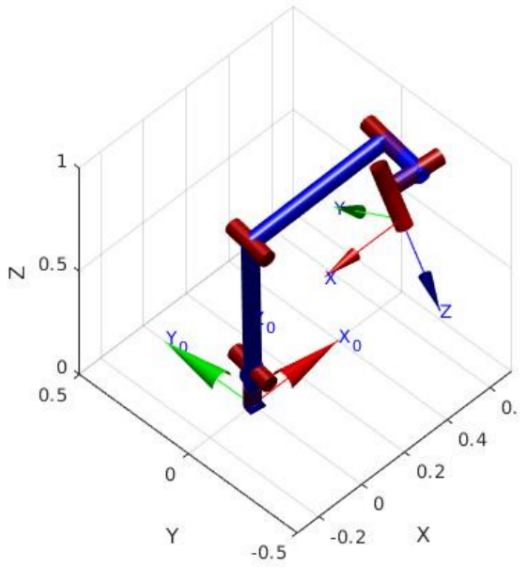


Figure 1: "Robotic Arm ur10e"

Let the frame of a fixed camera $\{C\}$ which is placed with an orientation $R_{0C} = I_3$, as shown in Figure 2. In the workspace of the arm is a sphere $r = 2cm$, the which due to its weight moves on a curved slide located at the level $x = 0.4$ of $\{0\}$. The contact frame $\{B\}$ is placed at the point of contact of the sphere with the slide, so that the $\bar{z}$ axis of the frame is perpendicular to the contact surface and the $\bar{y}$ axis is tangential.

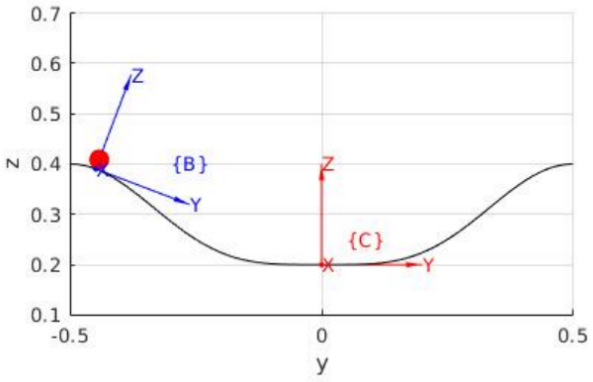


Figure 2: "Frames $\{B\}$ and $\{C\}$"

With the help of the camera, we can at any time determine the relative position of the sphere with respect to the frame of the camera p_{CB}, the translational speed v_{CB}, but also

the angular speed ω_{CB} . In the framework of the work, this happens with a class that simulates the movement of the sphere and returns the above 3 sizes.

At time 0 the joint values are $q_0 = [-0.140 \ -1.556 \ -1.359 \ 1,425 \ -1.053 \ -1.732]\, rad$, while the $\bar{y}$ axis of $\{B\}$ is in the $\begin{bmatrix} 0 & 0.9351 & -0.3543 \end{bmatrix}^T$ direction.

The tasks of this project are:

- First, determine the position and orientation of the end of the arm (g_{0e}).
- Then design a control signal $\dot{q}_r$, such that the tip tracks the movement of the arm with relative orientation $R_{BE} = Rot(y, 180°)$ and relative position $p_{BE} = \begin{bmatrix} 0 & 0 & 0.45 \end{bmatrix}^T$.

The following simulations were done in matlab using robotics toolbox, while the simulation step was fixed $T_s = 2ms$.

1.2 Task A

By importing into matlab the robotic arm model ur10e and then using robotics toolbox to solve the direct kinematics problem for the initial q_0 values of the joints, we obtain the following homogeneous transformation :

$$g_{0e}(0) = \begin{bmatrix} -1.0000 & 0.0003 & 0.0004 & 0.4 \\ 0.0001 & 0.8661 & -0.4999 & -0.2905 \\ -0.0005 & -0.4999 & -0.8661 & 0.8111 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1}$$

The above homogeneous transformation, and the initial values q_0 correspond to the following pose of the arm as seen in the Y-Z plane, with the tip located at $X = 0.4$:

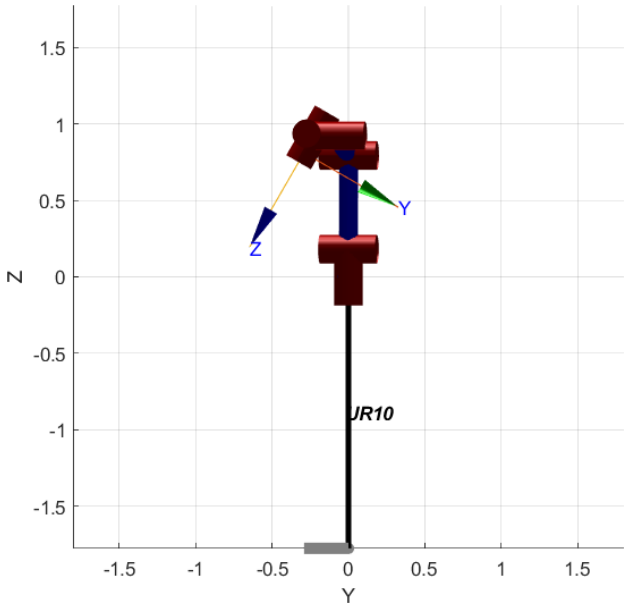


Figure 3: "Arm pose for $t = 0$"

1.3 Task B

1.3.1 Theoretical analysis

First we go to calculate the orientation of the frame {B} of the moving sphere with respect to the frame of the camera {C}. We know that the $\bar{y}_{CB}$ axis is in the direction $\begin{bmatrix}0 & 0.9351 & -0.3543\end{bmatrix}^T$. Also from figure 1 we see that $\bar{x}_{CB} = \begin{bmatrix}1 & 0 & 0\end{bmatrix}^T$. So calculating the cross product $\bar{x}_{CB} \times \bar{y}_{CB}$ we find the direction of the axis $\bar{z}_{CB} = \begin{bmatrix}0 & 0.3543 & 0.9351\end{bmatrix}^T$. Therefore it is:

$$R_{CB}(0) = \begin{bmatrix}1 & 0 & 0\\ 0 & 0.9351 & 0.3543\\ 0 & -0.3543 & 0.9351\end{bmatrix} \tag{2}$$

From the enunciation, but also from figure 2, we have $p_{0C} = \begin{bmatrix}0.4 & 0 & 0.2\end{bmatrix}^T$. Therefore:

$$g_{0C} = \begin{bmatrix}I_3 & p_{0C}\\ \mathbf{0} & 1\end{bmatrix}$$

So we can find the homogeneous transformation of the moving sphere {B} with respect to the inertial frame {0}:

$$g_{0B} = g_{0C}g_{CB} = \begin{bmatrix}I_3 & p_{0C}\\ \mathbf{0} & 1\end{bmatrix}\begin{bmatrix}R_{CB} & p_{CB}\\ \mathbf{0} & 1\end{bmatrix} \Rightarrow$$
$$g_{0B} = \begin{bmatrix}R_{CB} & p_{CB} + p_{0C}\\ \mathbf{0} & 1\end{bmatrix}$$

As mentioned in the introduction our goal is to design an appropriate control signal $\dot{q}_r$ such that the tip of the arm tracks the moving sphere with a desired relative bias and relative position. Let:

$$g_{BE_d} = \begin{bmatrix}R_{BE_d} & p_{BE_d}\\ \mathbf{0} & 1\end{bmatrix}$$
$$, R_{BE_d} = Rot(y, 180^\circ)\ , p_{BE_d} = \begin{bmatrix}0 & 0 & 0.45\end{bmatrix}^T$$

Therefore we can calculate the desired homogeneous transformation of the end {E} with respect to the inertial frame of the base of the arm {0} as follows:

$$g_d = g_{0B}g_{BE_d} = \begin{bmatrix}R_{CB} & p_{CB} + p_{0C}\\ \mathbf{0} & 1\end{bmatrix}\begin{bmatrix}R_{BE_d} & p_{BE_d}\\ \mathbf{0} & 1\end{bmatrix} \Rightarrow$$
$$g_d = \begin{bmatrix}R_{CB}Rot(y, 180^\circ) & R_{CB}p_{BE_d} + p_{CB} + p_{0C}\\ \mathbf{0} & 1\end{bmatrix}$$

Having now determined the desired position and orientation of the limb, we can design the control signal $\dot{q}_r$ based on the corresponding slides of the course dealing with kinematic control. So the speed commands result from the relation:

$$\dot{q}_r = J^{-1}(q)u \tag{3}$$

, where $u = \begin{bmatrix}\dot{p}_d\\ \omega_d\end{bmatrix} - K\begin{bmatrix}e_p\\ e_l\end{bmatrix}$, with $e_p = p_{0E} - p_d$ the position error and $e_l = \theta_e k_e$ the orientation error.

Let us now calculate $\dot{p}_d, \omega_d$. Starting with $\dot{p}_d$:

$$\dot{p}_d = \frac{d}{dt}(R_{CB}p_{BE_d} + p_{CB} + p_{0C}) \Rightarrow$$
$$\dot{p}_d = \dot{R}_{CB}p_{BE_d} + R_{CB}\dot{p}_{BE_d} + \dot{p}_{CB} + \dot{p}_{0C} \xrightarrow{p_{BE_d}, p_{0C}\text{fixed}}$$
$$\dot{p}_d = \dot{R}_{CB}p_{BE_d} + v_{CB} \tag{4}$$

In the above relation we can calculate $\dot{R}_{CB}$ from the relation:

$$\hat{\omega}_{CB} = \dot{R}_{CB}R_{CB}^T \Rightarrow$$
$$\dot{R}_{CB} = \hat{\omega}_{CB}R_{CB} \tag{5}$$

Similarly, we calculate ω_d:

$$\hat{\omega_d} = \dot{R}_d R_d^T \Rightarrow$$
$$\hat{\omega_d} = (\dot{R}_{CB}Rot(y,180^\circ) + R_{CB}\dot{Rot}(y,180^\circ))(R_{CB}Rot(y,180^\circ))^T \Rightarrow$$
$$\hat{\omega_d} = (\dot{R}_{CB}Rot(y,180^\circ) + R_{CB}\dot{Rot}(y,180^\circ))Rot^T(y,180^\circ)R_{CB}^T \Rightarrow$$
$$\hat{\omega_d} = \dot{R}_{CB}Rot(y,180^\circ)Rot^T(y,180^\circ)R_{CB}^T + R_{CB}\dot{Rot}(y,180^\circ)Rot^T(y,180^\circ)R_{CB}^T \Rightarrow$$
$$\hat{\omega_d} = \dot{R}_{CB}R_{CB}^T \Rightarrow$$
$$\hat{\omega_d} = \hat{\omega_{CB}} \text{ or } \omega_d = \omega_{CB} \tag{6}$$

, since from proposition 4.2 page 95 of the book $\dot{Rot}(y,\theta)Rot^T(y,\theta) = \dot{\theta}\hat{\bar{y}}$, so for a fixed angle 180° , it is $\dot{Rot}(y,180^\circ)Rot^T(y,180^\circ) = 0$.

1.3.2 Implementation in matlab

After first initializing the known quantities, the arm model and the moving sphere object, we implement the simulation with the help of an iterative loop. Each iterative step corresponds to $2ms$ of real time, which is essentially the sampling period of the system.

Our first operation inside the loop is to run the function sim_ball(T_s) to get $p_{CB}, v_{CB}, \omega_{CB}$. We then calculate, similar to the theoretical analysis, the orientation of the moving sphere {B} with respect to the camera frame {C}. Obviously, the $\bar{y}_{CB}$ axis which is always tangent to the surface where the sphere moves, will be in the same direction as v_{CB} which is also tangent to the surface. Therefore, by normalizing v_{CB} to become a unit vector we find $\bar{y}_{CB}$. Also, it is obvious from the figure that the axis $\bar{x}_{CB} = \begin{bmatrix}1 & 0 & 0\end{bmatrix}^T$. Thus by calculating the cross product $\bar{x}_{CB} \times \bar{y}_{CB}$ we find the axis $\bar{z}_{CB}$, so we have found R_{CB} and already having p_{CB}, from the camera we have found the homogeneous transformation g_{CB}.

Then, with the help of robotics toolbox, we solve the direct kinematics problem and calculate the transformation g_{0E}. Having calculated all the necessary variables, following the previous theoretical analysis, we calculate g_{0B}, g_d, $\dot{p}_d$, ω_d, e_p, e_l. Notice that no Unit Quaternions are used for the orientation error, but the equivalent rotation matrix operations are performed, and by converting to axis/angle equivalent we find e_l.

We now have all the necessary quantities to calculate the control input u and the Jacobian of the arm which we use to find the appropriate $\dot{q}_r$ in each cycle based on relation (3). All that remains is to impose the speed and acceleration limits of the joints. The rate limit is implemented with a simple saturation function (satq.m) that constrains $\dot{q}_r$ within the allowable limits for each joint. For the acceleration limit, a first-order discrete-time low-pass filter is used, which is applied to the signal $\dot{q}_r$ and is described by the relation:

$$y(n+1) = (1 - T_s f_c)y(n) + T_s f_c u(n),\ y(0) = u(0) \tag{7}$$

,where u the input signal, y the output signal, T_s the sampling frequency ($2ms$), f_c the cutoff frequency. From the above relation, we can easily form the derivative of the signal y ($\dot{y}(n) = \frac{y(n+1) - y(n)}{T_s}$) in the left member and get :

$$\dot{y}(n) = f_c(u(n) - y(n)) \tag{8}$$

Now let $u(n) \leq A$ and M be the limit to which we want to limit the absolute value of $\dot{y}(n)$. Therefore, we want to define f_c such that:

$$|\dot{y}(n)| \leq M \Rightarrow$$
$$|f_c(u(n) - y(n))| \leq M \Rightarrow$$

, but due to the upper bound of u it will be true:

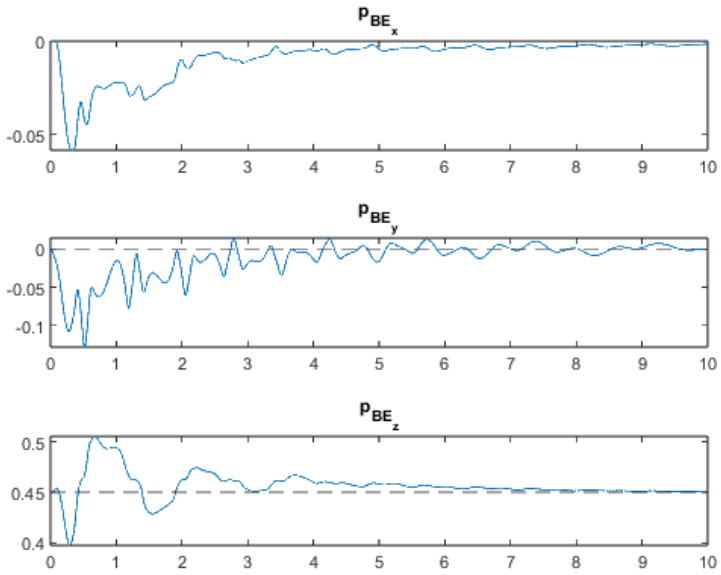
$$|(u(n) - y(n))| \leq 2A \Rightarrow$$
$$|f_c(u(n) - y(n))| \leq 2Af_c \Rightarrow$$
$$|\dot{y}(n)| \leq 2Af_c$$

So, it suffices to ensure that $2Af_c \leq M$. In order not to limit the filtered signal more than necessary, I choose the equality $2Af_c = M \Rightarrow f_c = \frac{M}{2A}$.

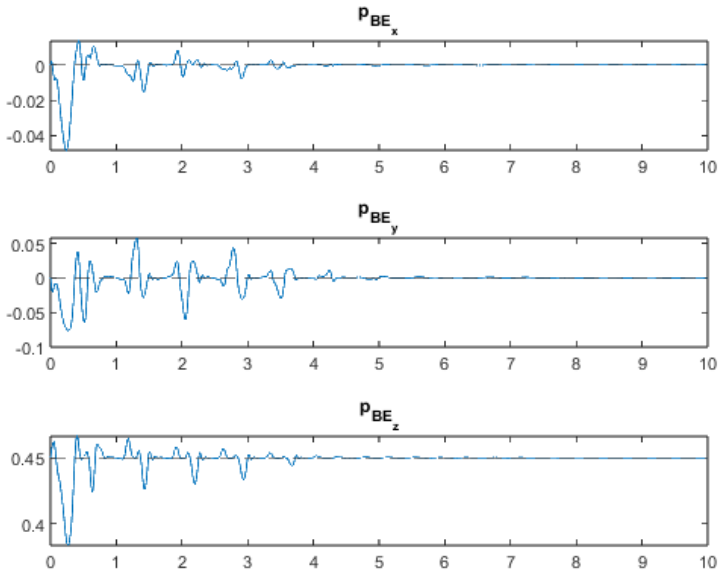
In our case the signal u is $\dot{q}_r$, y is the filtered $\dot{q}_r$, while $\dot{y}$ is $\ddot{q}_r$. So correspondingly the cutoff frequency $f_c = \frac{\ddot{q}_{lim}}{2\dot{q}_{lim}}$.

1.3.3 Simulation and Results

Starting the simulations, we first test the case where $k_p = k_l = I_3$, for which the diagram of the relative position p_{BE} is shown below:

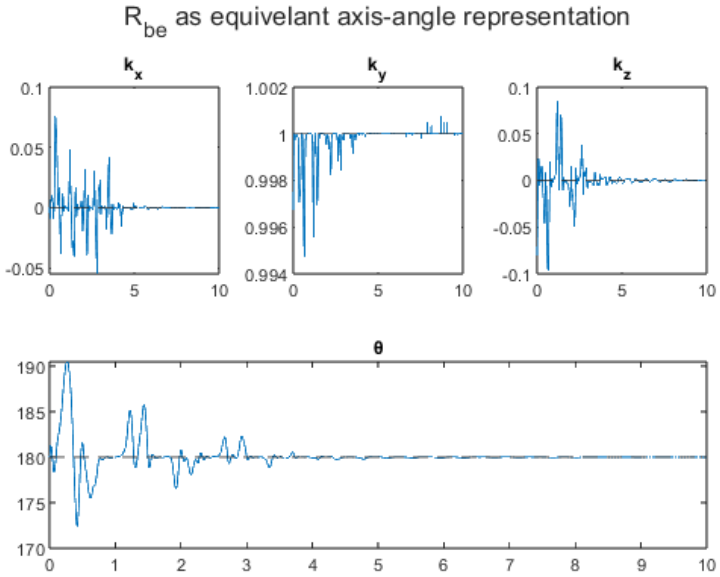


We observe that, indeed, over time the end of the arm achieves the desired relative position $p_{BE_d} = \begin{bmatrix} 0 & 0 & 0.45 \end{bmatrix}^T$, but we are not completely satisfied with the behavior of the system. Thus, after testing various values of the gains, looking for a good balance between convergence speed, maximum elevation and steady-state behavior, we arrive at the values $k_p = k_l = 100 I_3$. Here are the requested graphs, starting with the relative position p_B:

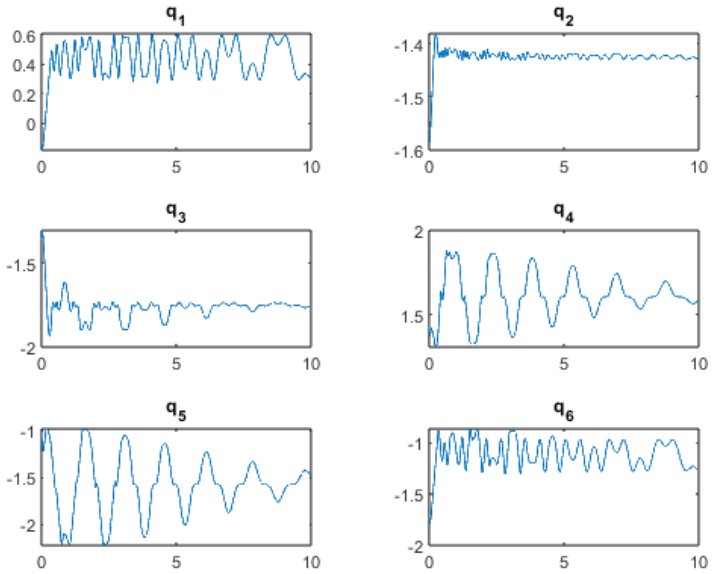


It is clear that in this case we have a faster speed of convergence to the desired position, while in fact the error e_p presents smaller values throughout, with the exception of the initial elevation of p_{BEz}. Notice that although the initial values are very close to the desired ones,

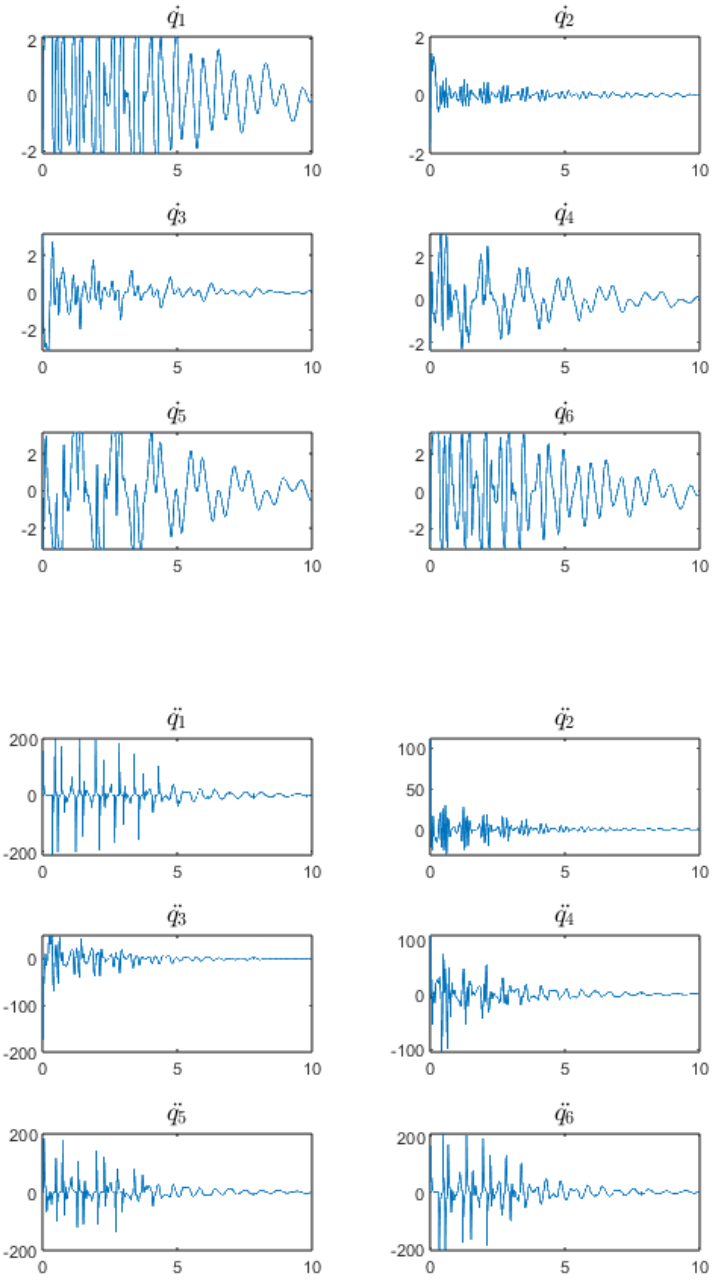
the zero initial velocity of the limb combined with the velocity and acceleration limits of the joints, lead to elevations at the beginning of the simulation. Here is the equivalent axis/angle diagram of the relative orientation R_{BE}:



Similar to the relative position, we see that initially there are some noticeable errors, but after a few seconds we converge to the desired orientation, and the errors become negligible. We can therefore conclude that we have achieved the control objectives (p_{BEd}, R_{BEd}). Below are the graphs of $q, \dot{q}, \ddot{q}$.



In full correspondence with the movement of the sphere and the limb which perform a damped oscillation, we observe that the displacements q of the joints, basically exhibit a periodic behavior with decreasing amplitude. Following are the graphs of $\dot{q}, \ddot{q}$ in which we can see that the velocity and acceleration limits have been successfully imposed, which are not exceeded for any joint and time point of the simulation.


$\dot{q}_1$
$\dot{q}_2$
$\dot{q}_3$
$\dot{q}_4$
$\dot{q}_5$
$\dot{q}_6$
$\ddot{q}_1$
$\ddot{q}_2$
$\ddot{q}_3$
$\ddot{q}_4$
$\ddot{q}_5$
$\ddot{q}_6$

2 Part B

2.1 Introduction

In part B of the task we want the robot to catch the moving ball. First we place in end of the robot the gripper shown in Figure 4. In the context of the work we consider that the framework of the gripper {T} is identical to the frame of the end {E}, so in the rest of the paper we will refer to the frame {E} and the end of the arm.

Figure 4: "Gripper"

More specifically, the goal is again to draw a signal $\dot{q}_r$, so that the arc of the arm approaches the sphere, resulting in a relative position $p_{BE} = \begin{bmatrix} 0 & 0 & 0.06 \end{bmatrix}^T$ and relative orientation $R_{BE} = Rot(y, 180°)$.

2.2 Theoretical Analysis - Implementation in matlab

Before we begin to approach the moving sphere, we will track its trajectory from a distance, with the desired orientation. Thus, in the initial stage of the simulation we will use the theoretical analysis and implementation done in Section A of the egasy so that the limb tracks the movement of the arm with a relative orientation $R_{BE} = Rot(y, 180°)$ and relative position $p_{BE} = \begin{bmatrix} 0 & 0 & 0.45 \end{bmatrix}^T$, until the error becomes sufficiently small and we approach the sphere.

We know that the sphere has a radius of $2cm$ so it will have a diameter and a maximum distance along the z axis from the surface of $4cm$. Since we want to approach the sphere with a relative distance $p_{BE_z} = 0.06$, it suffices that $\|e_p\| \leq 0.02$ while the orientation error is sufficiently small to avoid collisions with the sphere itself, but also with the surface on which it moves. Therefore, we choose $\|e_p\| abitmorestrictly \leq 0.01, \|e_l\| \leq 0.01$, as sufficient convergence criteria. In addition, for greater security and to confirm the stability of the system, we check whether the above conditions hold for 50 consecutive sampling periods (0.1 *seconds*), before starting the approximation.

As for the actual approximation of the sphere from the tip, it is achieved by plotting a linear trajectory with parabolic blending for p_{BE_z}. The trajectory can be calculated with the formulas in section 6.2 of the book, but in practice we use the robotics toolbox implementation, which takes over and makes it easier for us. So, to recap, we start exactly as in Part A, and when it is determined that the convergence criteria are met, we replace the constant p_{BE_d} with the trajectory we have designed.

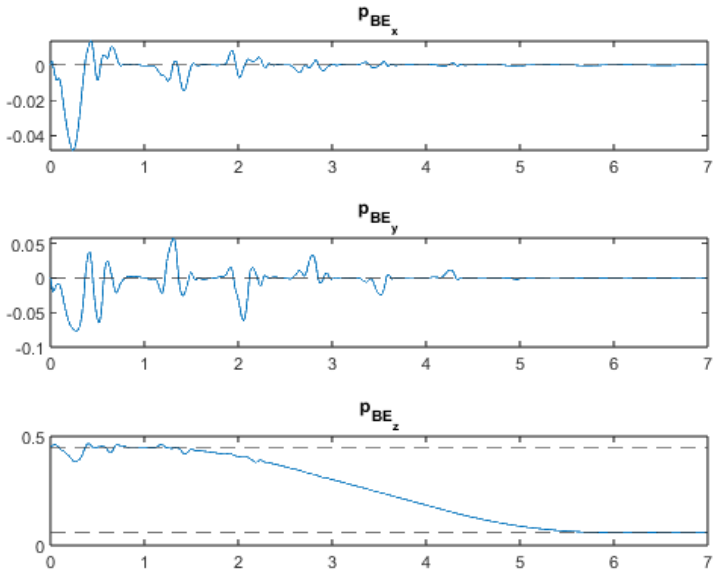
The previous theoretical analysis for the calculation of the control signal and the desired velocities is generally valid again, with the exception of the transport velocity $\dot{p}_d$ during the approach, since p_{BE_d} is no longer constant , and so its derivative is not zero. So during the approach the desired speed becomes:

$$\dot{p}_d = \dot{R}_{CB} p_{BE_d} + R_{CB} \dot{p}_{BE_d} + v_{CB} \tag{9}$$

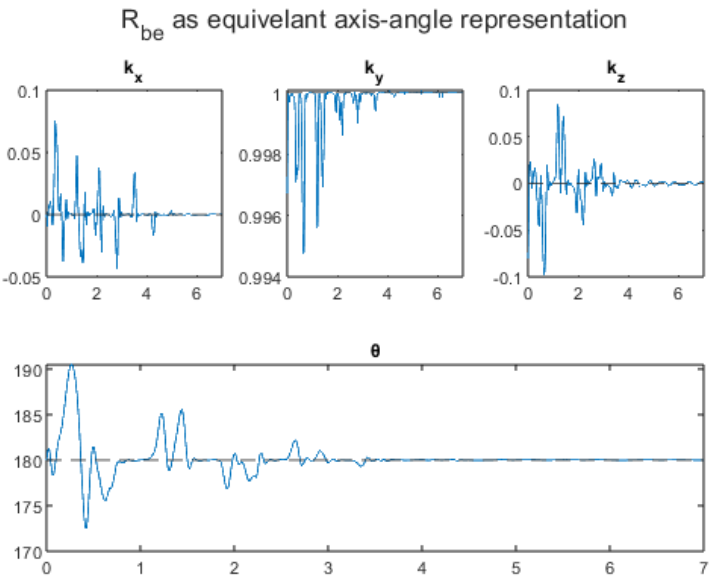
, which $\dot{p}_{BE_d}$ is known for each moment of the movement, since the trajectory is predetermined and designed by us.

2.3 Simulation and results

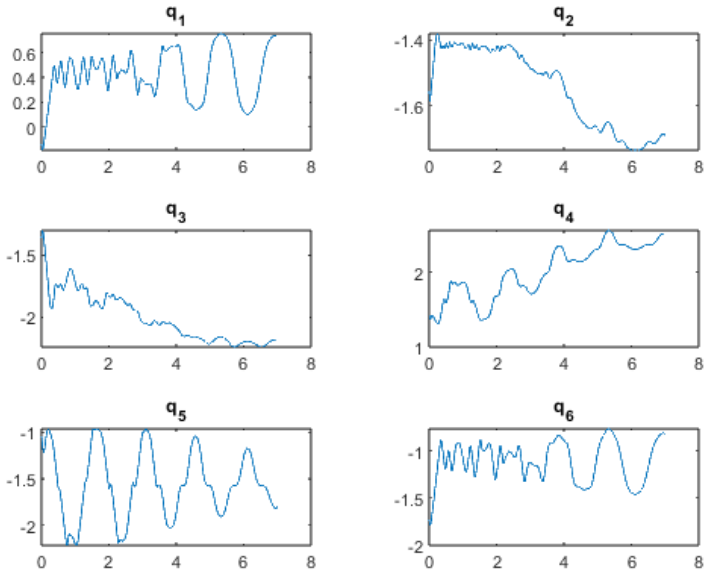
Moving on to the simulation, we choose the duration of the trajectory to be 5 seconds, so as not to have unwanted errors and elevations, and not to be limited by the speed and acceleration limits of the joints. As in Section A we use controller gains $k_p = k_l = 100 I_3$. Below are the simulation results, while running the matlab script partB.m you can see the 3d movement of the arm.



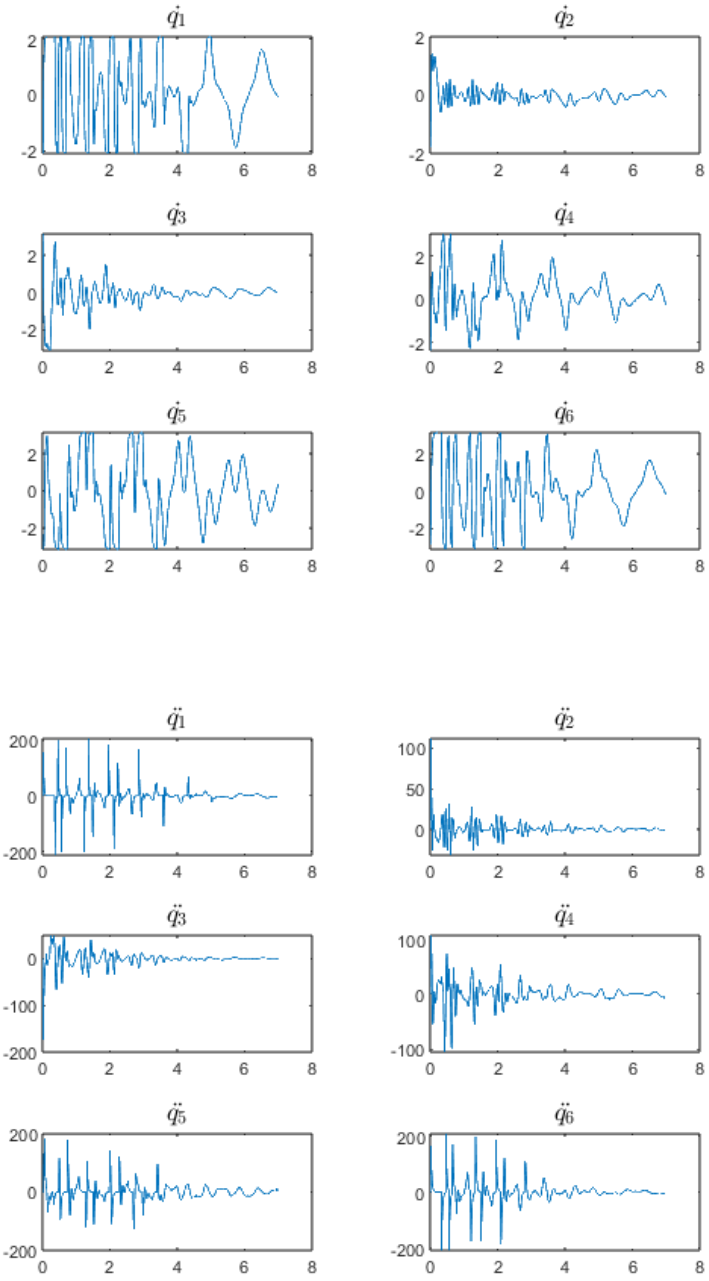
Starting with the relative position diagram p_{BE}, we notice that for the axes x, y the curves are identical to Section A. As for the distance p_{BE_z}, it follows the trajectory we drew without deviations, thus having a smooth course from the initial value of 0.45 to the final value of 0.06. In addition, we see that the criteria we set earlier are met for the 50th consecutive period at the time $t = 0.940\ s$, at which point the approach starts, is completed after 5 seconds, and stays in the desired relative position for 1 second, without difficulty and errors, making it possible to close the gripper fingers. Finally, since there are no elevations during the approach the grip never makes contact with the ball ($p_{BE_z} \leq 0.04$).



Observing the diagram of the relative orientation, we can easily conclude that the approach of the moving ball from the arm has not affected the orientation at all and is practically identical to Section A. This means that the end of the arm approaches the ball with the desired orientation and collision with the floor is avoided. The latter is best seen by watching the 3d movement of the arm.



The joint behavior is predictably different this time, but again periodicity occurs. To conclude, in the following graphs of $\dot{q}, \ddot{q}$ we can see how, in this case too, the speed and acceleration limits have been successfully imposed, which are not exceeded for any joint and moment of the simulation.


$\dot{q}_1$
$\dot{q}_2$
$\dot{q}_3$
$\dot{q}_4$
$\dot{q}_5$
$\dot{q}_6$
$\ddot{q}_1$
$\ddot{q}_2$
$\ddot{q}_3$
$\ddot{q}_4$
$\ddot{q}_5$
$\ddot{q}_6$