Final Project

Introduction to Robotics

Camryn Scully

May 10, 2023

To begin designing a trajectory for the LBR iiwa 7 R800, KUKA robot I first simplified the problem into two expressions.

$$T_T^0 = T_E^0\, T_C^E\, T_A^C\, T_T^A \tag{1}$$

$$T_R^0 = T_E^0\, T_R^E \tag{2}$$

Using forward kinematics, I determined the position and orientation of the target with respect to the base frame, represented by Equation 1. In addition, I found the pose of the rectangle attached to the end-effector of the robot also using forward kinematics. With this information, I implemented inverse kinematics to find the final joint configurations that place the rectangle on top of the target. With these joint angles, I planned a trajectory for the robot within the time, position, and velocity constraints and also accounted for obstacle avoidance. The DH parameters for the LBR iiwa 7 R800, KUKA robot are listed in Table 1.

Table 1. DH of the LBR iiwa 7 R800, KUKA

	d	θ	a	α
$0 \rightarrow 1$	0.340m	q_1	0	-90°
$1 \rightarrow 2$	0	q_2	0	90°
$2 \rightarrow 3$	0.400m	q_3	0	90°
$3 \rightarrow 4$	0	q_4	0	-90°
$4 \rightarrow 5$	0.400m	q_5	0	-90°
$5 \rightarrow 6$	0	q_6	0	90°
$6 \rightarrow$ E	0.126m	q_7	0	0

After computing forward kinematics to the end-effector using the above DH parameters, I computed the transformation from the end-effector to the camera. Once at the camera, I used the provided position and orientation of the Aruco marker with respect to the camera to develop a transformation matrix to the Aruco marker and then finally a simple translation to the target. All three of these matrices are below.

$$T_C^E = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & -0.0662 \\ 0 & 0 & 1 & 0.0431 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_A^C = \begin{bmatrix} 0.3211 & 0.9455 & 0.0541 & -0.1420 \\ 0.8238 & -0.3070 & 0.4765 & -0.0606 \\ 0.4671 & -0.1084 & -0.8775 & 0.3528 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_T^A = \begin{bmatrix} 1 & 0 & 0 & 0.104 \\ 0 & 1 & 0 & -0.104 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

With these transformations, I calculated the final transformation matrix from the base of the robot to the target, which describes the position and orientation of the target with respect to the base frame.

$$T_T^0 = \begin{bmatrix} 0.5236 & -0.8511 & -0.0386 & 0.0582 \\ 0.8512 & 0.5245 & -0.0166 & 0.5928 \\ 0.0344 & -0.0242 & 0.9991 & -0.0662 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Additionally, I found the transformation from the end-effector to the center of the rectangle. The frame is rotated so that the x-axis is aligned with the x-axis of the target frame.

$$T_R^E \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0.0455 \\ 0 & 0 & 1 & 0.06 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Prior to computing inverse kinematics with T0T, I rotated the z-axis by π to arrive at T0T2, which aligns the z- and y-axis with the rectangle's frame so that the rectangle can achieve the desired position and orientation of the target frame. I extracted the position and orientation of the target from T0T2 and then computed inverse kinematics in Matlab starting at the joint configuration, q_1 to arrive at the final joint configurations that place the rectangle shape on top of the target. Equation 1 displays the final joint configurations, q_f that I computed. To confirm that the robot does not exceed a position limit I added a constraint in the inverse kinematics for loop so that the q for each joint at each iteration can not be greater or smaller than the maximum and minimum position limit. If a value happens to be equal to or outside of one of the limits, I set that value equal to the minimum possible position limit to keep it within the range. This does not affect the trajectory of the robot, it only pertains to the q_f calculation to ensure those values are not outside of the provided joint limits.

$$q_f = \begin{bmatrix} -2.2934 & -2.0944 & -1.3761 & -1.1580 & -2.1666 & 1.5233 & -0.6205 \end{bmatrix} (rad) \quad (3)$$

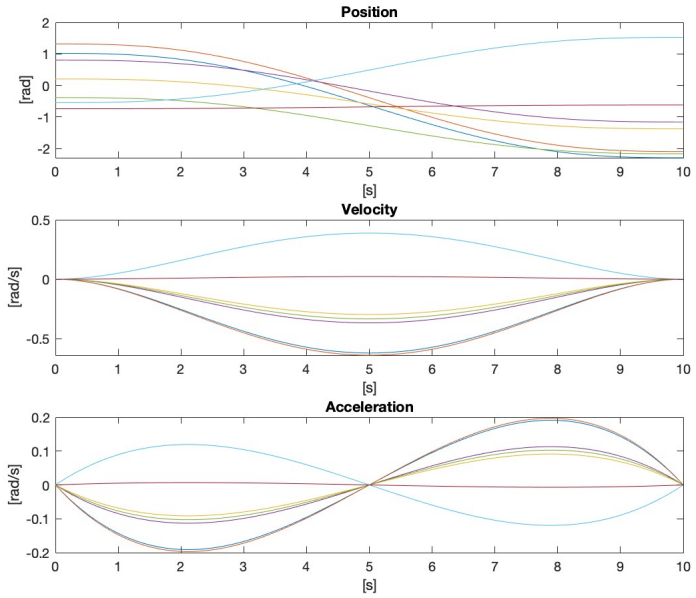
To compute a trajectory from q_1 to q_f, I represented the joint positions with a 5th order polynominal and computed both the position and velocity every 5 ms from time $t = 0$ to $t = 10$. Even though we're not provided an acceleration limit I also account for this so that it is equal to 0 at both $t = 0$ and $t = 10$, which is why I use the 5th order polynomial. I solved for the coefficients in Matlab and the position and velocity as functions of time are described

by Equation 2 and 3, respectively.

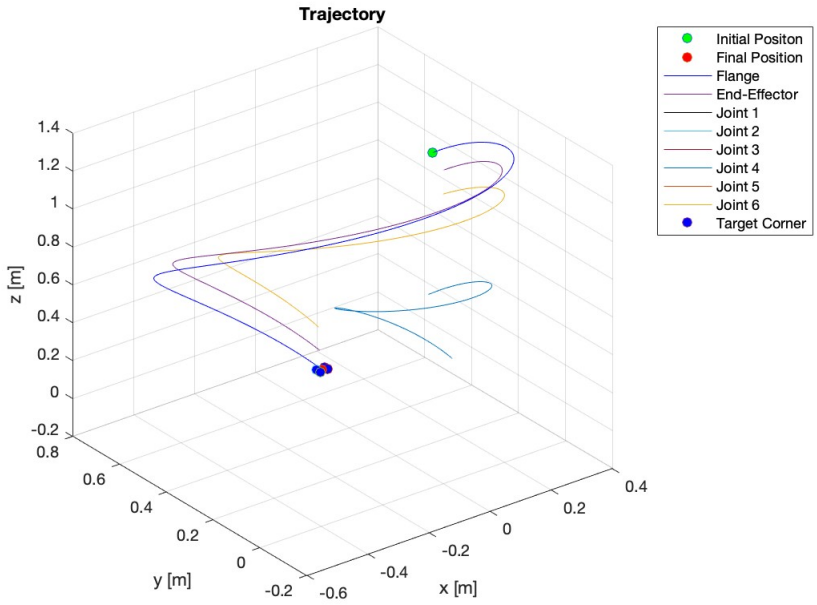
$$q(t) = a_5t^5 + a_4t^4 + a_3t^3 + a_2t^2 + q_i = 0.0012t^3 + q_1(i) \tag{4}$$

$$\dot{q}(t) = 5a_5t^4 + 4a_4t^3 + 3a_3t^2 + 2a_2t = 3(0.0012)t^2 \tag{5}$$

I plotted the position and velocity for the 10 seconds to confirm that no joint limits are exceeded and to export the position of each joint every 5 ms to generate the trajectory file that will be sent to the robot.



Additionally, to confirm that the robot does not hit the table holding the robot or the box that holds the target, I plotted the trajectory of each joint in operational space. The trajectory does not go below the z-coordinate of the target, -0.0662 [m] and the rectangle lands directly on the center of the target. The plot is below.


Trajectory
Initial Positon
Final Position
Flange
End-Effector
Joint 1
Joint 2
Joint 3
Joint 4
Joint 5
Joint 6
Target Corner
z [m]
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
y [m]
0.8
0.6
0.4
0.2
0
-0.2
x [m]
-0.6
-0.4
-0.2
0
0.2
0.4