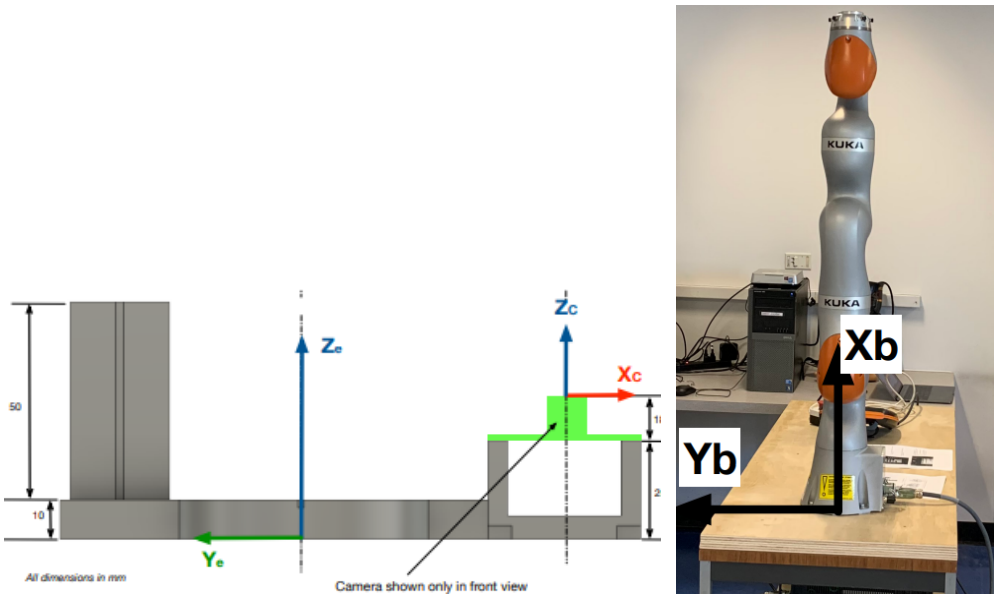
Vivek Dharmesh Mange
MEEG 671 Introduction to Robotics

Final project report

Introduction:

In this project we have demonstrated the trajectory generation for a 7 degree-of-freedom robot arm (LBR iiwa 7 R800, KUKA) robot with an adapter mounted at the end effector. The adapter includes a holding for a camera and the other side includes a rectangular shape design as shown in fig(1). The solution is defined in 3 steps, 1. Transformation calculation of the Target wrt base of the robot. 2. Calculation of inverse kinematics to find the final configuration of the robot when it reaches the target. 3. Trajectory generation reckons with the Pose and Velocity limitation of the joints. The script also generates the text file for the trajectory to test it on the actual robot.



fig(1) - Robot and object at the end effector

Solution:

1. Target Pose

Initially we calculate the transformation from base frame to target. To close the loop, we can describe the final T0T matrix as follows

$$T0T = T0E\_qc * TEC * TCA * TAT$$

We compute the forward kinematics of the robot at Q=Qc and get the transformation T0E_qc. We generate the TEC(End effector to camera) matrix based on the given data. To get the TCA(Camera to Aruko Marker) matrix, we use the eul2rotm function for ‘ZYX’ angles of the marker wrt to the camera. The TAT matrix representing the transform from Aruko marker to the Target has an Identity rotation matrix and offset

in x y z direction as per given information. After using the above equation we get the final pose and orientation of the Target wrt to the base of the robot.

2. Inverse Kinematics

After the calculation of the target pose and orientation, We perform inverse kinematics using jacobian. We compute transformation from base to end effector when the robot reaches the target. Assuming the object hit the target with expected pose, we use the below equation to calculate the T0E_qf.

$$T0E*TEF = T0F$$

$$T0F = T0T \quad \text{“based on the assumption that the object is at the target pose”}$$

$$T0E\_qf = T0T*inv(TEF)$$

Based on the above T0E_qf, we calculate the desired [X,Y,Z,phi,theta,psi] for the robot to reach the destination. We perform the multiple iterations of inverse kinematics calculation to reduce the error between the current pose of the robot and the destination pose. We use the below equations to compute the new q values using the analytical jacobian.

```
[xe,~] = fwd_kin(d, inv_q(:,i), a, alpha);
e(:,i) = [PD; PHID]-xe;
[Ja,~] = ana_jacob_calc(d, inv_q(:,i), a, alpha,Ta);
qdot = pinv(Ja)*K*e(:,i);
```

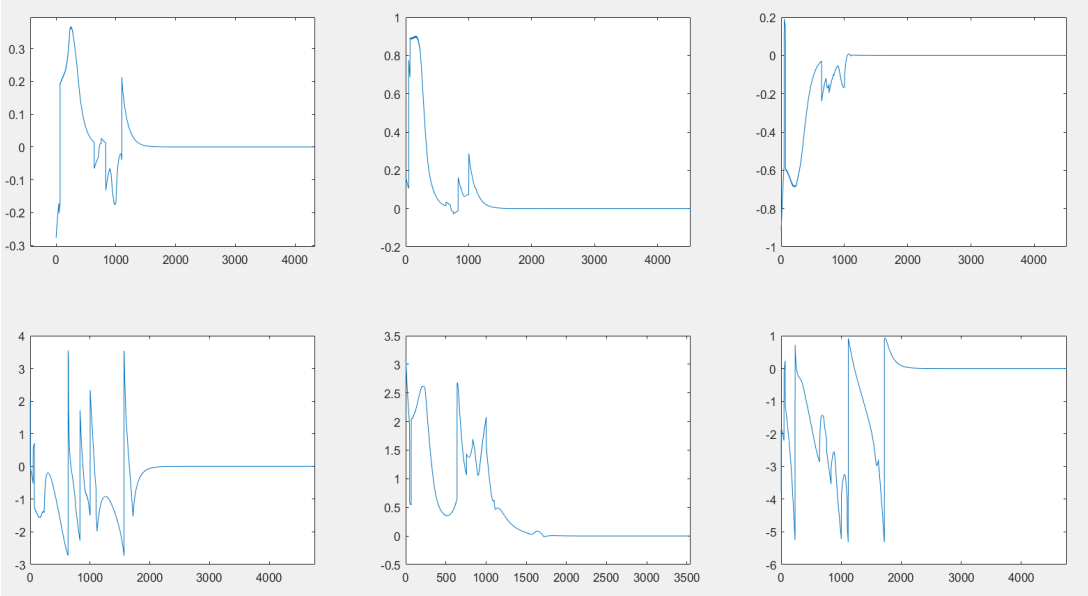


Fig (2) - Error plot

Where we have Xe as the current pose of the robot, e as the error between desired pose and current pose. To update the values of q, we use the analytical jacobian and update q in smaller increments of qdots. We also keep a track of the pose limits to generate the final configurations inside the limits. Based on the error we adjust the gain and try to converge it to 0.

3. Trajectory planning

After calculating the final configuration of the robot, we use 3 degree polynomial trajectory planning to generate a trajectory for all the 7 joints from q1 to qf. We use the below equation for q(t) and qdot(t). And we save values of q every 5 ms and for time instant [0, 10s].

$$q(t) = a_3*t^3 + a_2*t^2 + a_1*t + a_0$$
$$qdot(t) = 3a_3*t^2 + 2a_3 = -(1/15)*a_2$$
$$a_2*t + a_1$$

Based on the given conditions, at t=0, $a_0 = q(0)$, $qdot(0) = 0$ i.e $a_1 = 0$. To calculate a2 and a3, we use t= 10s,

$$q(10) = 1000a_3 + 100a_2 + 10a_1 + a_0$$
$$qdot(10) = 300a_3 + 10a_2 + a_1.$$

Solving the above equations, we get a relation with a3 and a2 as follows

$$a3 = -0.0667*a2$$

And when used on the q(t) at 10 sec we get a2 as follows

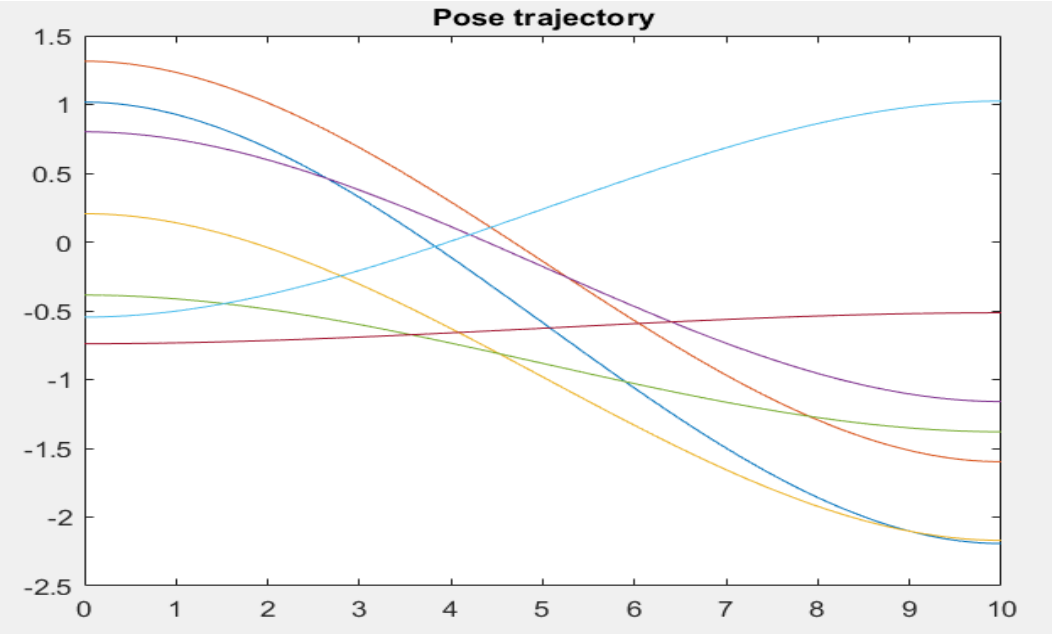
$$a_2 = (qf-qi)/33.300$$

We use the above constants in the trajectory equations and generate the trajectory for all the 7 joints and perform the limit check to verify the limits of the robot. Also generate a txt file with all the 2000 q values of the 7 joints which goes from q1 to qf. To verify the final transformation of the robot, we run forward kinematics to the final configuration of the trajectory planner and multiply the transformation of the end effector to the rectangle shape to get the final pose of the object and confirm it with the T0T matrix.

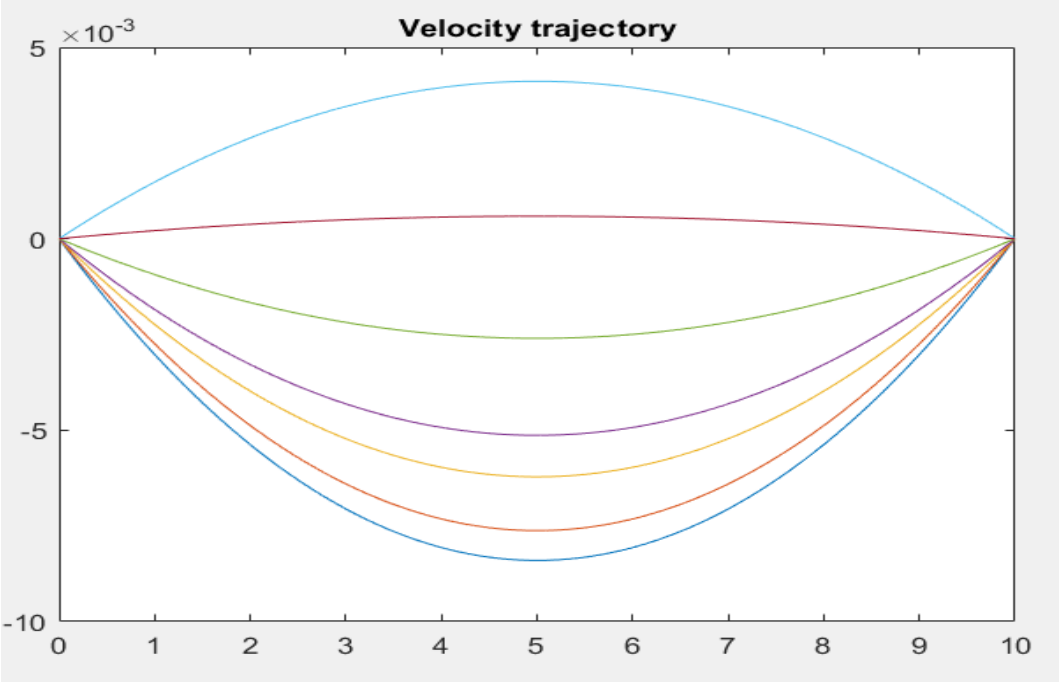
Vivek Dharmesh Mange
MEEG 671 Introduction to Robotics

Results:

I have attached the plot of pose and velocity trajectory wrt to time.



Pose Trajectory



Velocity Trajectory