MFET 348 – Advanced Industrial Robotics

Final project report

Name: Saurabh Singhal

Objective:

- The goal of this course project is to apply the knowledge and techniques learned in this class to solve real-world problems.
- The goal of my project is to build a simulation of an onion cutting robot using the concepts learned in the class.

Procedure:

1. Establish the reference frames of the robot and the counter on which the onion would be placed:

   I first assumed the robot's reference frame and then defined the counter's frame in the workspace of the robot. The origin of the robot's reference frame was attached to the base of the robot and the origin of the counter was attached to the lower left corner of the counter.

2. Define the transformation matrix from the origin of the robot frame F_R to the origin of the counter frame F_C:

   After establishing the reference frames for the robot and the counter, I defined the position vector of the counter's origin with respect to the robot's frame origin which was

   ```
   p = [0.381; -0.1524; 0.6604];
   ```

   After establishing the position vector, I established the rotation matrix of the counter's origin with respect to the robot's frame origin which was

   ```
   R = [1 0 0; 0 1 0; 0 0 1];
   ```

   The transformation matrix was then developed by combining p and R as follows

   ```
   T = [R p];
   T = [T; [0 0 0 1]]
   ```

3. Establish the path (points) to cut the onion:

   I defined the points to create a path to cut the onion on the counter top with respect to the counter's reference frame. I assumed the onion to be a hemisphere of diameter 3 inches and the onion was at the center of the counter as shown in Figure 1. I also assumed the counter to be 0.3048 m x 0.1524 m in dimension.

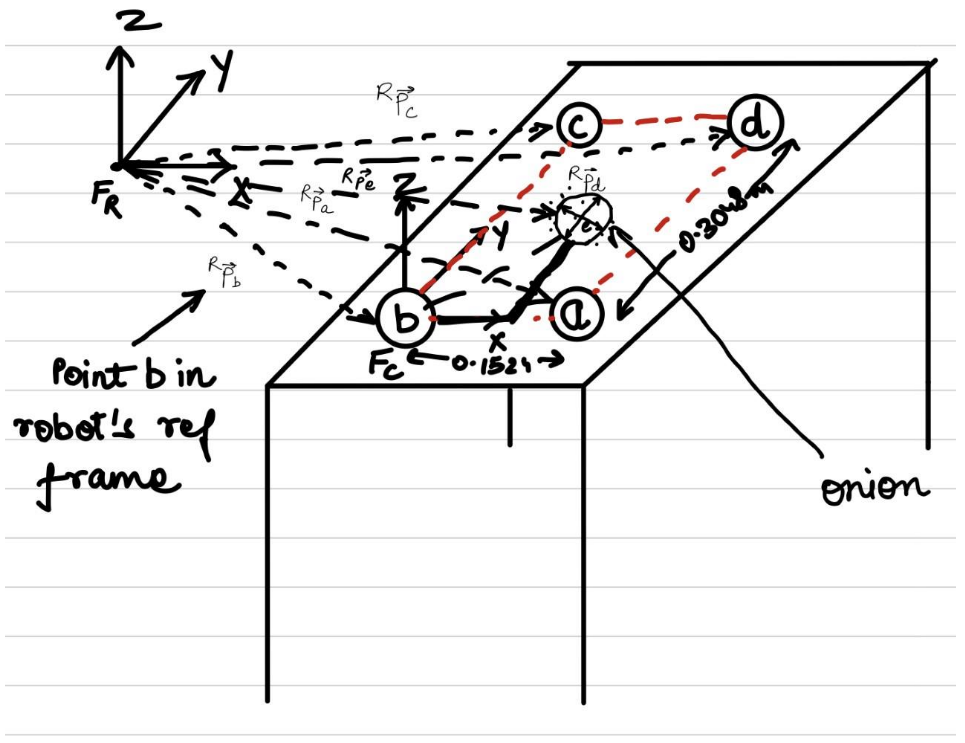

Figure 1: The onion on the counter with respect to the robot's reference frame

To divide the onion in 6 parts, I had to define 5 cuts and their path in form of points with respect to counter's reference frame which are:

```
% Path to cut the onion in Fc frame
A = zeros(4, 20);
%1st cut
A(:, 1) = [0.0381; 0.127; 0.0508; 1];
A(:, 2) = [0.0381; 0.127; 0; 1];
A(:, 3) = [0.1143; 0.127; 0; 1];
A(:, 4) = [0.1143; 0.127; 0.0508; 1];
%2nd cut
A(:, 5) = [0.0381; 0.1397; 0.0508; 1];
A(:, 6) = [0.0381; 0.1397; 0; 1];
A(:, 7) = [0.1143; 0.1397; 0; 1];
A(:, 8) = [0.1143; 0.1397; 0.0508; 1];
%3rd cut
A(:, 9) = [0.0381; 0.1524; 0.0508; 1];
A(:, 10) = [0.0381; 0.1524; 0; 1];
A(:, 11) = [0.1143; 0.1524; 0; 1];
A(:, 12) = [0.1143; 0.1524; 0.0508; 1];
%4th cut
A(:, 13) = [0.0381; 0.1651; 0.0508; 1];
A(:, 14) = [0.0381; 0.1651; 0; 1];
A(:, 15) = [0.1143; 0.1651; 0; 1];
A(:, 16) = [0.1143; 0.1651; 0.0508; 1];
%5th cut
A(:, 17) = [0.0381; 0.1778; 0.0508; 1];
A(:, 18) = [0.0381; 0.1778; 0; 1];
```

```
A(:, 19) = [0.1143; 0.1778; 0; 1];
A(:, 20) = [0.1143; 0.1778; 0.0508; 1];
```

The trajectory was plotted and is shown in Figure 2:

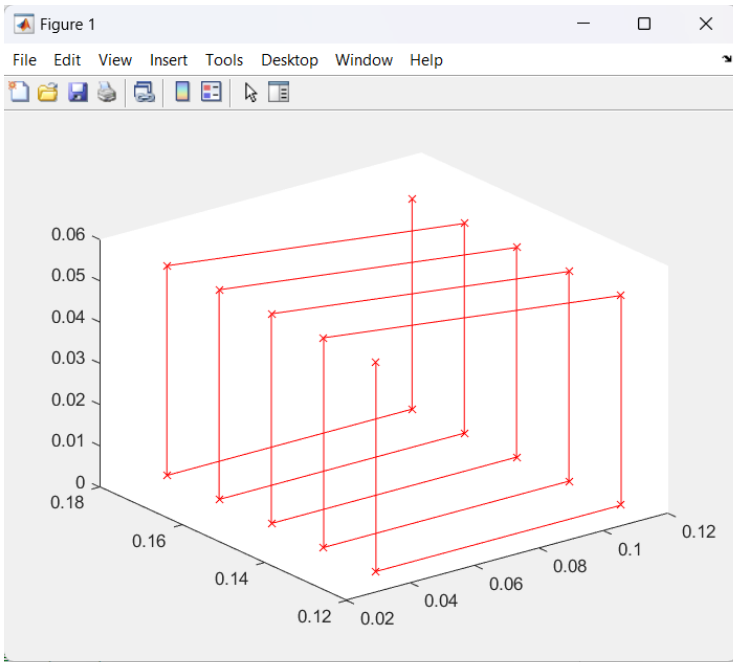

Figure 2: The trajectory to be followed by the end-effector of the robot

4. Converted the points in counter's reference frame to the robot's reference frame and then created the transformation matrices representing the points in the robot's reference frame:

```
B = zeros(4, 20);
B = T*A

TB = zeros(4, 4, 20);


% the rotation maxtrix representing the orientation
% of the end-effector
Rr = [1 0 0; 0 1 0; 0 0 1; 0 0 0];

% Replace the "orientation area" of the transformation matrices with the
% above contructed rotation matrix.
for i = 1:1:20
TB(:,:,i) = [Rr B(:,i)];
end
```

5. Trajectory planning in cartesian space:

   The trajectory of the end effector was planned in the cartesian space by inserting four points between the two adjacent points.

```
% Trajectory planning in Cartesian space.
```

```
n = 4;
TP = zeros(4, 4, n*20);
% For points defining the path, insert n points between two adjacent points
for i = 1:1:19
      TP(:,:,((i-1)*n+1):i*n) = ctraj(TB(:,:,i), TB(:,:,i+1), n)
end
```

6. Inverse kinematics to identify the joint angles:

   The inverse kinematics for the robot were solved to identify the joint angles so that the end effector could follow the proposed trajectory.

   ```
   q = p560.ikine(TP);
   ```

7. Plot the robot motion and the trajectory simultaneously:

   To make it look like the robot is actually cutting the onion, the robot's motion and the points were plotted simultaneously.

   ```
   % Bring up a figure window, clean up, setup viewpoint and axis ranges
   figure(2)
   clf;
   xlim([-1,1]);
   ylim([-1,1]);
   zlim([-1,1]);
   view([105,14]);

   % Plot the points while moving the robot so that it
   % looks the robot is cutting a vegetable
   for i = 1:1:(n*20)
         plot2(transpose(TP(1:3,4,i)),'r.')
         p560.plot(q(i,:))
         hold on
   end

   % Move robot to the vertical "ready" or "home" configuration after finish
   % writing
   p560.plot(qr)
   ```

Analysis and conclusion:

Through this project I got to revisit the concepts learnt in this class and got to apply them to solve some real-world problems. To make this project successful I used spatial description and transformation to define the counter's transformation matrix with respect to robot's base. Defined the points' transformation matrices in robot's reference frame using forward kinematics. Planned the trajectory of the robot's end-effector using trajectory planning in cartesian space, and found robot angles using inverse kinematics. I learnt how to apply the knowledge gained in the class in the real world and I look forward to use it further in my career.