FL. 2018 ESE 447.02 Robotics Lab

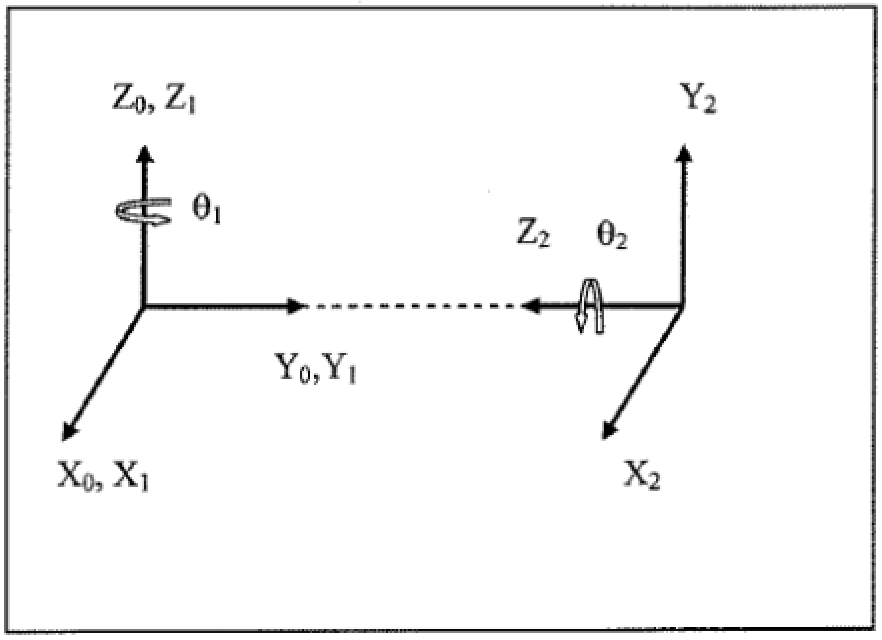
Lab Journal 6

10/14/2018

1. Exercise 3. Objectives.

   Use Matlab to create a 2-axis robot simulation model, find the robot's work envelope and use the Quanser Pendulum to test the simulation.

   The schematic of the frames of the Quasnser pendulum is shown below.



   The transfer matrices are given below.

$$ {}^0_1T = \begin{bmatrix} Cos(\theta_1) & -Sin(\theta_1) & 0 & 0 \\ Sin(\theta_1) & Cos(\theta_1) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^1_2T = \begin{bmatrix} Cos(\theta_2) & -Sin(\theta_2) & 0 & 0 \\ 0 & 0 & -1 & (l_1 + l_2) \\ Sin(\theta_2) & Cos(\theta_2) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

2. Task 1. Use 'plot3' command in Matlab to draw base of robot in Frame-O (the stationary frame).

Plot3 is the embedded function in Matlab to plot in 3 dimensional coordinate system. Suppose we have two points in the space whose coordinates are (x1,y1,z1) and (x2,y2,z2), to plot the points and line them up, we can simply use the command plot3([x1,x2],[y1,y2],[z1,z2]). The 3-D space can be stabilized into certain length by fixing the axis. Here we used the line: axis([-15 15 -15 15 -15 15]) to limit and fix all the three axis from -15 to 15.

3. Draw first link of the robot with the following points in Frame-1: (0,0,0), (0,0,-1), (0,6,-1), (0,6,0) and (0,8,0).
   As stated above, the links were drawn using the line:
   plot3([0,0,0,0,0],[0,0,6,6,8],[0,-1,-1,0,0],'-r*');
   Where '-r*' defines the color and the point type.
4. Task 3. Write a function in MATLab to convert the points representing the first link. The homogeneous transformation matrix is given above. This transformation will introduce the variable 'theta1' which is the rotation about the Z-axis of FRAME-1. Your program should accept input values for 'theta1' and plot the results to the figure in Task #1. Loop program in order to create animation effect.
   The Matlab function to present the transformation is shown below.
   ```
   function [ newp,x_points,y_points,z_points ] = trans( theta,A,B,C,D )%The input arguments are the angles and the points
       t=[cos(theta) -sin(theta) 0 0;
      sin(theta) cos(theta) 0 0;
      0 0 1 0;
       0 0 0 1];
       points = [A;B;C;D];
       newp = t*points;
   %     x_points=newp(1:1,1:5);
   %     y_points=newp(2:2,1:5);
   %     z_points=newp(3:3,1:5);
   end
   ```
   The input arguments are the theta1 (here we called theta), which is the angle rotated by the first link, and the points representing the first link.
   The function uses the given matrix and calculates the values inside the matrix, then outputs the points after the transformation. The format of output is a matrix containing the coordinates. To call the points, we can just use the bottom three lines that were commented out.
   To plot the animation effect, we used a while loop. The initial theta value is 0 and it iterates for each loop. The theta value increases by 0.1 for every iteration, and the end points are plotted for every iteration and held in the plot. The loop stops when the theta value reaches the input theta value.
   The graph is representing the base frame since the axes are fixed at the base frame. The work envelope for this one link robot is a circle since the link is rotating 2 dimensionally in a surface.
5. Task 4. Draw second link of robot with the following points in FRAME-2: (0,0,0) and (0,12,0). These points represent the link as would be viewed from FRAME-2.
   Similar to task 3, the second link was drawn by connecting the two points, the only difference was that we had to convert the points viewed from FRAME-2 to FRAME-1, which means we had to use the transformation matrix from FRAME-1 to FRAME-2.