

DH parameter(clearer images are provided in the image folder)

```
DH Parameters are
[[0 theta1 pi/2 33.3]
 [0 theta2 -pi/2 0]
 [8.8 0 -pi/2 31.6]
 [-8.8 theta4 pi/2 0]
 [0 theta5 pi/2 38.4]
 [8.8 theta6 -pi/2 0]
 [0 theta7 0 -20.7]]
```

Jacobian Matrix(clearer images are provided in the image folder)

```
neha@neha-Lenovo-IdeaPad-S540-15IML-D: ~/Documents/sem1/modelling/nmarne_hw4
[-8.8 theta4 pi/2 0]
[0 theta5 pi/2 38.4]
[8.8 theta6 -pi/2 0]
[0 theta7 0 -20.7]]
jacobian_matrix is
31.6*sin(theta1)*sin(theta2) + 8.8*sin(theta1)*sin(theta2)*sin(theta2 - theta1) - 20.7*sin(theta1)*sin(theta2)*cos(theta2 - theta1) - 20.7*sin(theta1)*sin(theta2 - theta1)*cos(theta2 - theta1) + 38.4*sin(theta1)*sin(theta2 - theta1) - 8.8*sin(theta1)*cos(theta2 - theta1) - 8.8*sin(theta1)
-20.7*sin(theta1)*sin(theta2)*sin(theta2 - theta1) - 8.8*sin(theta1)*sin(theta2)*cos(theta2 - theta1) - 31.6*sin(theta2)*cos(theta2 - theta1) - 8.8*sin(theta2)*sin(theta2 - theta1)*cos(theta2 - theta1) + 20.7*sin(theta2)*cos(theta2 - theta1)*cos(theta2 - theta1) + 20.7*sin(theta2 - theta1)*cos(theta2 - theta1)*cos(theta2 - theta1)
0
0
0
1
cos(theta1)*cos(theta2)*cos(theta2 - theta1) + 8.8*sin(theta1)*cos(theta2 - theta1) - 20.7*sin(theta1)*sin(theta2)*cos(theta2 - theta1) - 8.8*sin(theta1)*cos(theta2)*cos(theta2 - theta1) (-8.8*sin(theta2 - 20.7*sin(theta2)*sin(theta2 - theta1)*cos(theta2 - theta1) - 8.8*sin(theta2)*cos(theta2 - theta1) - 8.8
38.4*sin(theta2 - theta1)*cos(theta2 - theta1) + 8.8*cos(theta2)*cos(theta2 - theta1) + 8.8*cos(theta2)*cos(theta2 - theta1)*cos(theta2 - theta1) - 8.8*cos(theta2)*cos(theta2 - theta1) (-8.8*sin(theta2 - 20.7*sin(theta2)*sin(theta2 - theta1)*cos(theta2 - theta1) - 8.8*sin(theta2)*cos(theta2 - theta1) - 8.8
-31.6*sin(theta2) - 8.8*sin(theta2)*sin(theta2 - theta1) + 20.7*sin(theta2)*cos(theta2 - theta1)*cos(theta2 - theta1)

sin(theta2 - theta1)*cos(theta2)*cos(theta2 - theta1) + 8.8*sin(theta2 - theta1) - 31.6*cos(theta2) + 20.7*cos(theta2)*cos(theta2 - theta1) + 38.4*cos(theta2 - theta1)*cos(theta2) (20.7*sin(theta1)*sin(theta2 - theta1)*cos(theta2 - theta1) + 8.8*sin(theta1)*cos(theta2 - theta1) + 8.8*sin(theta2 - theta1)*cos(theta2 - theta1) + 8.8*sin(theta2 - theta1)
sin(theta2 - theta1)*cos(theta2)*cos(theta2 - theta1) + 8.8*sin(theta2 - theta1) - 31.6*cos(theta2) + 20.7*cos(theta2)*cos(theta2 - theta1) - 38.4*cos(theta2 - theta1)*sin(theta2) (20.7*sin(theta1)*sin(theta2 - theta1)*cos(theta2 - theta1) + 8.8*sin(theta1)*cos(theta2 - theta1) + 8.8*sin(theta2 - theta1)
+ 20.7*sin(theta2 - theta1)*cos(theta2 - theta1) + 38.4*sin(theta2 - theta1) + 8.8*cos(theta2) + 8.8*cos(theta2)*cos(theta2 - theta1) - 8.8*cos(theta2 - theta1) 8.8*sin(theta1)*sin(theta2 - theta1) - 20.7*sin(theta1)*cos(theta2 - theta1) - 20.7*sin(theta2)
sin(theta1)
-cos(theta1)
0
)*cos(theta1)*cos(theta2 - theta1) - 8.8*sin(theta2 - theta1) - 20.7*cos(theta2)*cos(theta2 - theta1) + 38.4*cos(theta2 - theta1)*cos(theta2) -20.7*sin(theta1)*sin(theta2)*cos(theta2 - theta1) - 8.8*sin(theta1)*cos(theta2)*cos(theta2 - theta1) - 20.7*sin(theta1)*sin(theta2)*cos(theta2 - theta1) -
)*cos(theta1)*cos(theta2 - theta1) - 8.8*sin(theta2 - theta1) - 20.7*cos(theta2)*cos(theta2 - theta1) + 38.4*cos(theta2 - theta1)*sin(theta2) -20.7*sin(theta1)*sin(theta2)*sin(theta2)*cos(theta2 - theta1) - 8.8*sin(theta1)*sin(theta2)*cos(theta2 - theta1) + 20.7*sin(theta1)*cos(theta2 - theta1)*cos(theta2 - theta1)
- theta1)*cos(theta2) + 38.4*sin(theta2 - theta1) - 8.8*cos(theta2)*cos(theta2 - theta1) + 8.8*cos(theta2 - theta1) -(20.7*sin(theta1) + 8.8*cos(theta1))*sin(theta2)*sin(theta2 - theta1)
sin(theta2)*cos(theta2) -sin(theta2 - theta1)*cos(theta1)
sin(theta1)*sin(theta2) -sin(theta1)*sin(theta2 - theta1)
cos(theta2) cos(theta2 - theta1)
8.8*sin(theta1)*cos(theta2)*cos(theta2 - theta1) 8.8*sin(theta1)*sin(theta2)*sin(theta2 - theta1) - 20.7*sin(theta1)*sin(theta2)*cos(theta2 - theta1) - 20.7*sin(theta1)*sin(theta2 - theta1)*cos(theta2 - theta1) - 8.8*sin(theta1)*cos(theta2)*cos(theta2 - theta1) - 8.8*sin(theta2 - theta1)*co
)*cos(theta2) + 8.8*cos(theta2)*cos(theta2 - theta1) -20.7*sin(theta1)*sin(theta2)*sin(theta2 - theta1) - 8.8*sin(theta1)*sin(theta2)*cos(theta2 - theta1) - 8.8*sin(theta1)*sin(theta2 - theta1)*cos(theta2 - theta1) + 20.7*sin(theta1)*cos(theta2 - theta1)*cos(theta2 - theta1) -
8.8*sin(theta1)*cos(theta2)*cos(theta2 - theta1) 8.8*sin(theta1)*sin(theta2)*sin(theta2 - theta1) - 20.7*sin(theta1)*sin(theta2)*cos(theta2 - theta1) - 20.7*sin(theta1)*sin(theta2 - theta1)*cos(theta2 - theta1) - 8.8*sin(theta1)*cos(theta2)*cos(theta2 - theta1) - 8.8*sin(theta2 - theta1)*co
i)*cos(theta2) + 8.8*cos(theta2)*cos(theta2 - theta1) -20.7*sin(theta1)*sin(theta2)*sin(theta2 - theta1) - 8.8*sin(theta1)*sin(theta2)*cos(theta2 - theta1) - 8.8*sin(theta1)*sin(theta2 - theta1)*cos(theta2 - theta1) + 20.7*sin(theta1)*cos(theta2 - theta1)*cos(theta2 - theta1) -
-8.8*sin(theta1)*sin(theta2 - theta1)*cos(theta2 - theta1) + 20.7*sin(theta1)*cos(theta2 - theta1) + 20.7*sin(theta2 - theta1)*cos(theta2 - theta1) + 8.8*cos(theta2)*c
sin(theta1)*cos(theta2) + sin(theta1)*cos(theta1)*cos(theta2 - theta1)
sin(theta1)*sin(theta2)*cos(theta2 - theta1) - cos(theta1)*cos(theta2)
sin(theta2)*sin(theta2 - theta1)
sin(theta2)*sin(theta2 - theta1)

s(theta1)*cos(theta2) + 20.7*cos(theta1)*cos(theta2)*cos(theta2 - theta1) 0
8.8*sin(theta1)*sin(theta2)*cos(theta1) + 20.7*sin(theta1)*cos(theta1)*cos(theta2) 0
cos(theta2 - theta1) 0
(sin(theta1)*sin(theta2) - cos(theta2)*cos(theta2 - theta1))*sin(theta2) - sin(theta2 - theta1)*cos(theta1)*cos(theta2)
-(sin(theta1)*cos(theta2)*cos(theta2 - theta1) + sin(theta2)*cos(theta1))*sin(theta2) - sin(theta1)*sin(theta2 - theta1)*cos(theta2)
-sin(theta2)*sin(theta2 - theta1)*cos(theta2) + cos(theta2)*cos(theta2 - theta1)
```

Method2 Used for setting up the jacobian

$$T_{i-1}^i = \begin{bmatrix} \cos\theta_i & -\cos\alpha_i \sin\theta_i & \sin\alpha_i \sin\theta_i & a_i \cos\theta_i \\ \sin\theta_i & \cos\alpha_i \cos\theta_i & -\sin\alpha_i \cos\theta_i & a_i \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Transformation matrix of the above form was used for each frame transfer of frame

Z_i = first 3 elements of 3rd column of T_i^0

With $Z_0 = [0 \ 0 \ 1]$

X_p = first 3 aspects of 4th column of T_n^0

Jacobian =

$${}^0J = \begin{bmatrix} \frac{\partial {}^0X_p}{\partial q_1} & \dots & \frac{\partial {}^0X_p}{\partial q_i} & \dots & \frac{\partial {}^0X_p}{\partial q_n} \\ {}^0Z_1 & \dots & {}^0Z_i & \dots & {}^0Z_n \end{bmatrix}$$

The equation used for the circle of motion

$$y^2 + (z - 72.5)^2 = 100$$

Velocity Equation

$$\begin{aligned} y_{\text{dot}} &= 4.0 * \pi * \sin(\theta) \\ z_{\text{dot}} &= 4.0 * \pi * \cos(\theta) \end{aligned}$$

Inverse Kinematics Equation

$$\dot{q} = J^{-1}(q) * \dot{X}$$

With

$$\dot{X} = [v_x, v_y, v_z, \omega_x, \omega_y, \omega_z]^T$$

Joint angles were calculated using

$$q_{next} = q_{current} + \dot{q}_{next} \Delta t$$

Using these Joint angles positions of final end effector was obtained with following the equation of FK

$$X = T_n^0 * q$$

This X was plotted using Matplot

Theta being updated by $2 * \pi / 40$ (since I decided to sample the circle into 40 point) at each iteration for 5 sec

Final output

