Robotics

Name:
1.Laila Tarek
2.Miran Samer
3. Mariam Eladawy
4.Abduallah Nizar

links	beta	d	a	alpha
1	θ1	0	0	+90
2	θ2	0	L1	0
3	θ3	0	L2	0
4	θ4	0	l3	0

- **Forward Kinematics:**

X=cos θ1(L2cos θ2+L3cos (θ2+ θ3)+L4cos (θ2+ θ3+ θ4))

Y= sinθ1(L2cos θ2+L3cos (θ2+ θ3)+L4cos (θ2+ θ3+ θ4)

Z=L1+L2 sinθ2+L3 sin (θ2+ θ3)+L4sin (θ2+ θ3+ θ4)

- **Full_Transformation:**

Rot(Z, θ1)+Trans(0,0,0,0)+Trans(0,0,0,0)+Rot(X,90)

Rot(Z, θ2)+Trans(0,0,0,0)+Trans(L1,0,0,0)+Rot(X,0)

Rot(Z, θ3)+Trans(0,0,0,0)+Trans(L2,0,0,0)+Rot(X,0)

Rot(Z, θ4)+Trans(0,0,0,0)+Trans(L3,0,0,0)+Rot(X,0)

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

Transformation Matrices

◆ A_1 (Link 1, α = +90°)

$$A_1 = \begin{bmatrix} \cos\theta_1 & 0 & \sin\theta_1 & 0 \\ \sin\theta_1 & 0 & -\cos\theta_1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

◆ A_2 (Link 2)

$$A_2 = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & L_1\cos\theta_2 \\ \sin\theta_2 & \cos\theta_2 & 0 & L_1\sin\theta_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

◆ A_3 (Link 3)

$$A_3 = \begin{bmatrix} \cos\theta_3 & -\sin\theta_3 & 0 & L_2\cos\theta_3 \\ \sin\theta_3 & \cos\theta_3 & 0 & L_2\sin\theta_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

◆ A_4 (Link 4)

$$A_4 = \begin{bmatrix} \cos\theta_4 & -\sin\theta_4 & 0 & L_3\cos\theta_4 \\ \sin\theta_4 & \cos\theta_4 & 0 & L_3\sin\theta_4 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

◆ Overall Forward Kinematics

$$T_0^4 = A_1 A_2 A_3 A_4$$

- **Inverse Kinematics:(**[Robotics System Toolbox - MATLAB - MathWorks]**)**

```
import numpy as np
T1 = 0
T2 = 0
T3 = 0
T4 = 0   #The angle values are T1-T6 that are taken as 0 (zero) degrees for testing.
T5 = 0
T6 = 0
T1=(T1/180.0)*np.pi
T2=(T2/180.0)*np.pi
T3=(T1/180.0)*np.pi
T4=(T2/180.0)*np.pi
T5=(T2/180.0)*np.pi
T6=(T2/180.0)*np.pi
R0_1 =[[np.cos(T1) , 0 , np.sin(T1)] ,[np.sin(T1) , 0 , -np.cos(T1)] ,[0,1,0]]
R1_2 =[[np.cos(T2) , -np.sin(T2) , 0] ,[np.sin(T2) , np.cos(T2) , 0] ,[0,0,1]]
R2_3 =[[np.cos(T3) , -np.sin(T3) , 0] ,[np.sin(T3) , np.cos(T3) , 0] ,[0,0,1]]
R3_4 =[[-np.sin(T4) , 0 , np.cos(T4)] , [np.cos(T4) , 0 , np.sin(T4)] ,[0,1,0]]
R4_5 = [[0,0,1] , [-np.sin(T5) , -np.cos(T5) , 0] ,[np.cos(T5) , -np.sin(T5) , 0]]
R5_6 =[[np.cos(T6) , -np.sin(T6) , 0] ,[np.sin(T6) , np.cos(T6) , 0] ,[0,0,1]]
R0_2 = np.dot(R0_1 ,R1_2)
R0_3 = np.dot(R0_2 , R2_3)
R0_4 = np.dot(R0_3 , R3_4)
R0_5 = np.dot(R0_4 , R4_5)
R0_6 = np.dot(R0_5 , R5_6)
print(R0_6)
```

- ### **Gripper:**

Simulation Environment

We used **CoppeliaSim** to simulate the robotic system with a focus on testing and controlling the **gripper mechanism**.

The simulation environment allowed us to implement and validate the gripper control code, including opening, closing, and object grasping operations.

By using CoppeliaSim, we were able to visualize the interaction between the gripper and objects, and evaluate grasp stability in real time.

This helped us debug the control logic, avoid hardware damage, and ensure reliable gripper performance before real-world deployment.