CSE461: Introduction to Robotics

LAB REPORT 04(Kinematics)

Group: 05

Name: Md. Mostafijur Rahman

Student Id: 22101721

Section: 03

G-Suite: md.mostafijur.rahman@g.bracu.ac.bd

Date Of Submission: 09.09.2024

Objectives of kinematics:

- Determine Position and Orientation: To determine the end effector of the robot's position and orientation with respect to a fixed reference coordinate system.
- Path Planning: To develop the path or trajectory that the end effector of the robot must follow in order to arrive at the desired location and position.
- Control Movement: To control the movement of each joint in order to obtain the required end effector motion.
- Collision Avoidance: To ensure that the robot moves in a way that avoids colliding with obstacles in its environment.

Code and output (Inverse Kinematics):

Code:

```python
import math

l1,l2=48,23
x,y,z=10.00, 28.00, 29.00

d1=math.sqrt(x**2+y**2)
th1=math.acos(x/d1)

d2=math.sqrt(d1**2 + z**2)
th7=math.atan(z/d1)


th3 =math.acos((l1**2 + l2**2 - d2**2)/ (2*l1*l2))

th6= math.acos((l1**2+d2**2-l2**2)/(2*l1*d2))

th2=th7+th6

print(f"theta 1: {math.degrees(th1)}")
print(f"theta 2: {math.degrees(th2)}")
print(f"theta 3: {180 - math.degrees(th3)}")
```

Output:

```
theta 1: 70.3461759419467
theta 2: 72.90608160716421
theta 3: 120.11992665856022
```

Explanation of formula used in the code (Inverse Kinematics):

The code calculates the joint angles (theta1, theta2, and theta3) for a 3-DOF robotic arm given the arm segment length (l1, l2) and target position (x, y, z) in Cartesian coordinates.d1 represents the horizontal projection of the target position onto the XY plane. It's the distance from the origin to the target point's projection on the XY plane.

d1 = math.sqrt(x2 + y**2)**

This is found using the Pythagorean theorem: d1= $\sqrt{x^2 + y^2}$.

Then calculates base rotation angle :

th1 = math.acos(x / d1)

th1 is the angle of the base rotation required to reach the target position in the XY plane.This angle is calculated using the inverse cosine function.It determines how much the arm needs to rotate about the base to align with the target's projection on the XY plane.

Calculate d2 the total distance from the base to the target

d2 = math.sqrt(d12 + z**2)**

Calculate th7 elevation angle from XY Plane to the Target.

th7 is the angle between the horizontal projection d1 and the vertical distance z. This is the elevation angle that elevates the arm from the horizontal XY plane to the level of the target point.

th3 = math.acos((l12 + l2**2 - d2**2) / (2 * l1 * l2))**

Calculate angle (Using Cosine Law)

th6 = math.acos((l12 + d2**2 - l2**2) / (2 * l1 * d2))**

th6 is the angle between the first arm segment l1 and the line connecting the base to the target d2 .

th2 = th7 + th6

th2 is the sum of th7 elevation from the XY plane) and th6(angle of the first arm segment with respect to d2).This angle represents the overall elevation of the shoulder joint required to position the arm towards the target.

Difference between Forward and Inverse Kinematics

Forward kinematics involves calculating the position and orientation of the robot's end effector based on given joint parameters. Inverse kinematics involves calculating the joint parameters required to achieve a specified position and orientation of the robot's end effector.

Forward kinematics is a straightforward calculation since the joint parameters are known, and the end effector position is calculated using the robot's kinetic equations.. Inverse kinematics is more complicated because it may involve solving nonlinear equations and may have multiple solutions or no solution for a given end-effector position.

Discussion on when to use Forward and Inverse kinematics:

Forward Kinematics determines the location and orientation of a robot's end-effector based on known joint angles and link lengths. When the joint configurations are predefined, it is utilized in control, simulation, trajectory planning, verification, and stability analysis.On the other hand, inverse kinematics determines the joint angles required to accomplish a particular end-effector position and orientation. When accurate positioning is needed, it's crucial for task execution, interactive applications, motion planning, user interfaces, and virtual worlds.While inverse kinematics is more complicated and frequently necessitates iterative methods due to the possibility of many solutions or constraints, forward kinematics is simpler and depends on geometric relationships.Use inverse kinematics to position the robot's joints to attain a desired objective, and forward kinematics to forecast the robot's location from supplied joints.