Robotic arm
3-DOF

ABOUT US

My Project is A 3-DOF (Degrees of Freedom) robotic arm is a simple robot with three joints that can move. Each joint lets the arm move in a different way, such as up and down, side to side, or turning. This means the arm can work in 3D space.

These robotic arms are often used in factories, schools, and labs because they are easy to understand and useful. Even though they only have three joints, they can do many tasks like picking up things, drawing shapes, or putting parts together.

The design of the arm is similar to a human arm, so it's great for learning how robots move and how to control them using math and programming

EQUATIONS

FORWARD PART

In the forward part:

Let:
$\theta_1, \theta_2, \theta_3$: Joint angles (input)
L_1, L_2, L_3 : Link lengths
T01: *Transformation from base frame to joint 1 (rotation about Z and translation along Z)*

T01 =
[cos(θ1) -sin(θ1) 0 0 ;
sin(θ1) cos(θ1) 0 0 ;
0 0 1 L1 ;
0 0 0 1]

- Rotation around Z-axis by angle O1
- Translation along Z by L1 (vertical displacement)

% T12: Transformation from joint 1 to joint 2 (rotation about Y and translation along X & Z)

T12 =
[cos(θ2) 0 sin(θ2) L2*cos(θ2) ;
0 1 0 0 ;
-sin(θ2) 0 cos(θ2) L2*sin(θ2) ;
0 0 0 1]

T23: Transformation from joint 2 to joint 3 (rotation about Y and translation along X & Z)

T23 =
[cos(θ3) 0 sin(θ3) L3*cos(θ3) ;
0 1 0 0 ;
-sin(θ3) 0 cos(θ3) L3*sin(θ3) ;
0 0 0 1]

- Similar structure to T12

Forward Kinematics:

T02 = T01 * T12;
T03 = T02 * T23;

position Vectors:

O0 = [0; 0; 0]; % Base
O1 = T01(1:3,4); % Position of Joint 1
O2 = T02(1:3,4); % Position of Joint 2
O3 = T03(1:3,4); % End-effector position

- x = O3(1): X-coordinate of the end-effector.
- y = O3(2): Y-coordinate of the end-effector.
- z = O3(3): Z-coordinate of the end-effector.

INVERSE PART

1. Radial Distance (Horizontal Projection)

r = sqrt(px^2 + py^2);

This calculates the distance from the robot’s base to the projection of the target point on the XY plane.

2. Adjusted Vertical Distance

z = pz - L1;

Subtract the fixed height of the first link (L1) to compute the vertical distance relevant to the two main arm segments.

1. **3. Check Reachability with Cosine Law**

D = (r^2 + z^2 - L2^2 - L3^2) / (2 * L2 * L3);

If |D| ≤ 1, the point is reachable; otherwise, it is out of the robot’s range.

4. Compute Elbow Angle (θ3)

```
theta3 = atan2(sqrt(1 - D^2), D);
```

5. Compute Shoulder Angle (θ2)

```
theta2 = atan2(z, r) - atan2(L3 * sin(theta3), L2 + L3 * cos(theta3));
```

.

6. Compute Base Rotation (θ1)

```
theta1 = atan2(py, px);
```

dynamics:

- I calculated angular velocity for each joint using finite difference:
- $\omega = (\theta_i - \theta_{i-1}) / dt$
- Then, I computed torque by multiplying the moment of inertia by the angular acceleration:
- $\tau = I \times (\omega_i - \omega_{i-1}) / dt$
- This process was done individually for each joint, allowing dynamic torque calculation across the entire motion.

DRAW CYLINDER

In 3D, for any vector v = [vx, vy, vz]^T, we define the skew-symmetric matrix as:

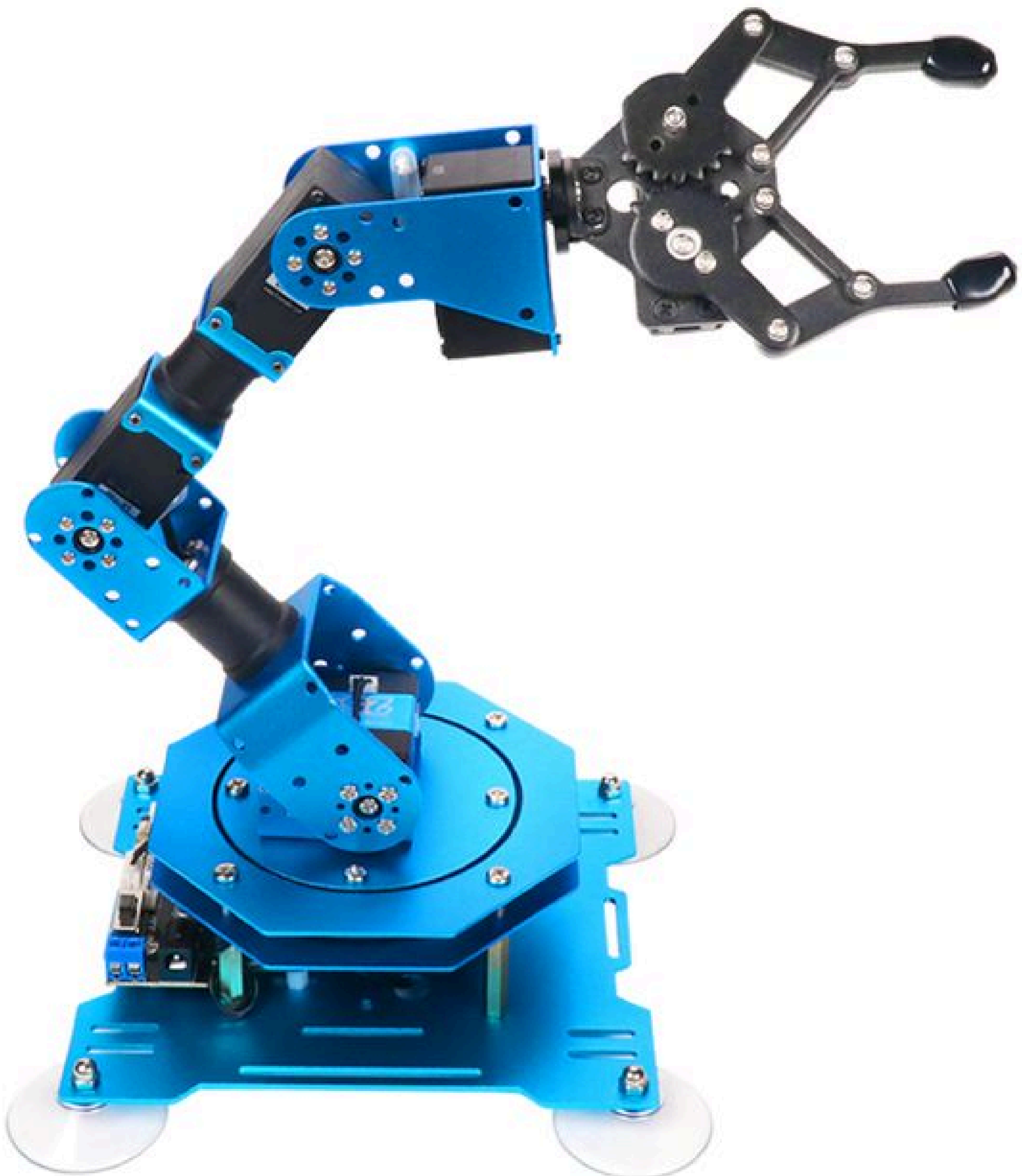
[v]_× =
[0 -vz vy
vz 0 -vx
-vy vx 0]

Rodrigues' Rotation Formula: R = I + [n]_× * sin(θ) + ([n]_×)^2 * (1 - cos(θ))

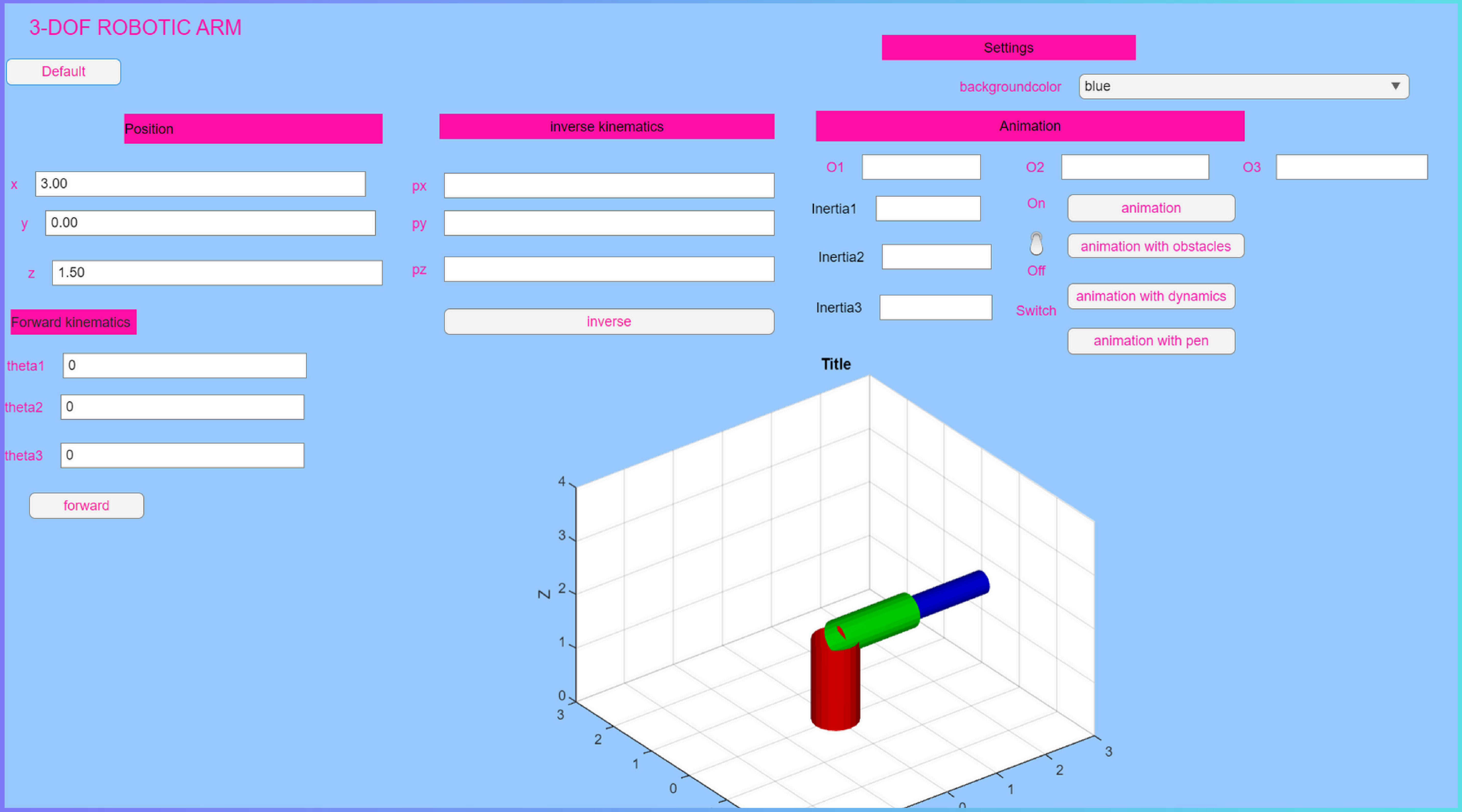
Where:

- R: rotation matrix
- I: identity matrix
- [n]_×: skew-symmetric matrix of axis n
- θ: angle of rotation

. i use this formula to Rotate each point of cylinder: p_rotated = R * p + p1



My appdesigner:



Main functions :

- *Forward kinematics*:This calculates the position of the robot arm’s end based on the joint angles
- Inverse kinematics:This finds the joint angles needed to reach a specific position in space
- *Animation*:This shows how the robot arm moves step by step as the angles change.
- *Animation with dynamics*:This includes real-world forces like gravity, speed, and torque to make the robot's movement look more realistic and physically accurate.

- **Animation with obstacles:**

In this mode, the robot moves in an environment with obstacles. If a collision happens during the motion, the system detects it and displays an error message to alert the user.

- **Default function:**

This function sets the robot's starting position to O1 = 0, O2 = 0, O3 = 0 by default, before any movement or animation begin

- **Change background color function:**

This function allows the user to change the background color of the appdesigner.

THYNK UNLIMITED

THANK YOU