KULLIYYAH OF ENGINEERING

DEPARTMENT OF MECHATRONICS ENGINEERING

PROJECT REPORT:
ROBOTICS PROJECT

FUNDAMENTALS OF ROBOTICS

MCTA 3332

SEMESTER 2, 2024/2025

SECTION 1

LECTURER'S NAME:

PROF. DR. TANVEER SALEH

PREPARED BY:

NO	NAME	MATRIC NO.
1.	NOR MAISARAH BINTI ISMAIL	2213080

TABLE OF CONTENTS

PROBLEM STATEMENT

This project involves analysing the motion of a 3DOF SCARA robot using Forward Kinematics (FK) and Inverse KInematics (IK) approaches. The goal is to model and stimulate joint movements using Simulink/Simscape and to compare the result with analytical solutions to evaluate accuracy and robot behavior.

OBJECTIVES

1. To develop and analyse the forward kinematics of a SCARA robot using simulation.
2. To simulate periodic joint inputs and collect joint data (position, velocity, etc.).
3. To verify simulation results against analytical FK.
4. To perform IK analysis for a closed-loop path.
5. To compare analytical IK vs simulated IK solutions.

METHODOLOGY

Robot Parameters

- Degrees of Freedom = 3 (Revolute-Revolute-Prismatic)
- Base height = 100cm
- L1 = 80cm
- L2 = 60cm
- Vertical displacement, D3 = 50cm

The robot has no wrist joint. The motion is considered in the X-Y plane for revolute joints and along the Z-axis for the prismatic joint.

Forward Kinematics (FK)

Analytical

The Denavit-Hartenberg (D-H) method was used to derive the transformation matrices for the FK model of the RRP SCARA robot. The transformation from the base to the end-effector was obtained using:

$$T = T1*T2*T3$$

Where:

T1 represents the transformation from the base to link 1,

T2 from link 1 to link 2,

T3 from link 2 to the prismatic axis.

Symbolic variables θ1, θ2, and D3 were defined in MATLAB, and the end-effector position (x, y, z) was derived.

Simulation

In Simulink with Simscape Multibody:

- Inputs: Periodic sine wave functions were applied.
- Simulation setup:
    - θ1 = [-60,60]
    - θ2 = [-60,60]
    - D3 = [0,-50]
- The robot was modeled using revolute and prismatic joints connected with rigid transforms.
- XY-Axis Graph, Scope blocks and "Simulation Data Inspector" were used to visualize joint data over time.

Inverse Kinematics (IK)

A closed-loop path was defined in 3D space with constant Z (height). The target end-effector positions along the path were sampled at discrete time steps.

Analytical

- The inverse kinematics was solved using geometric/trigonometric methods:

$$\theta 2 \;=\; cos^{-1}\left(\frac{x^2 + y^2 - L1^2 - L2^2}{2 \times L1 \times L2}\right)$$

$$\theta 1 \;= tan^{-1}\left(\frac{y}{x}\right) \;-\; tan^{-1}\left(\frac{L2 \times sin(\theta_2)}{L1 + \; L2 \times cos(\theta_2)}\right)$$

- Using known x, y positions to solve for θ1 and θ2:
- The vertical height (z-axis) directly gives D3.

Simulation

- Computes the required trajectory.
- Sum Blocks
    - Calculate the error between desired joint values and actual joint feedback.
- PID Controllers
    - Used to minimize error for each joint.
    - Output torque/force values to drive the robot joints.

- SCARA Robot Model (Simscape Multibody)
    - Receives joint control signals and simulates motion.
    - Outputs actual joint positions.
- Forward Kinematics Block
    - Converts joint angles back for comparison with the desired trajectory.
- Feedback Loop
    - Actual joint positions fed back into the Sum block to compute the next error.
    - Output joint variables from simulation were compared with analytical IK solutions to evaluate accuracy.

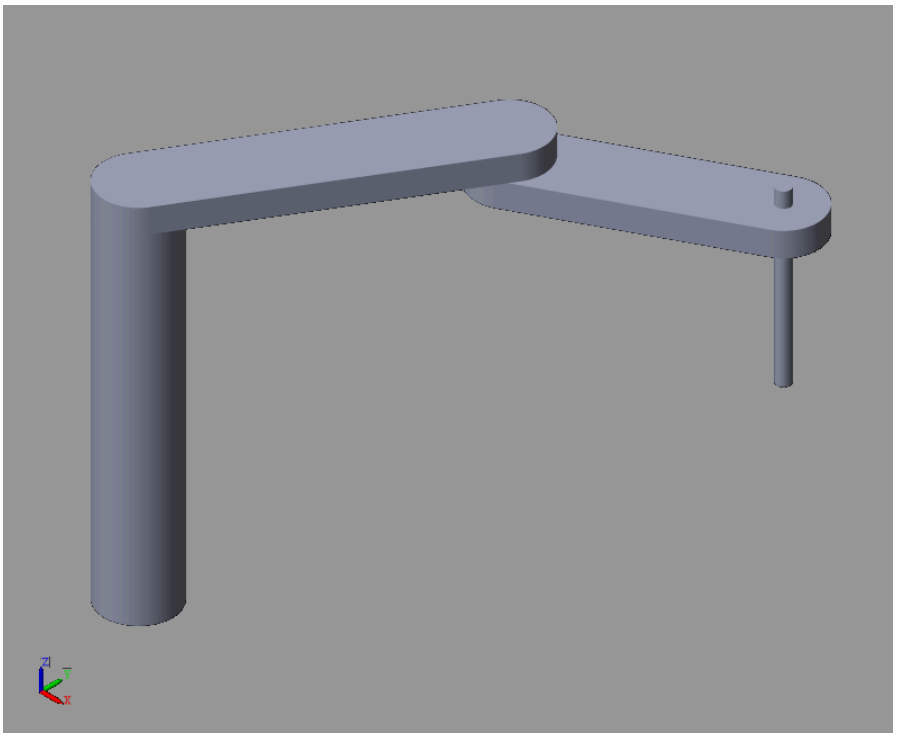
Comparison and Validation

For both FK and IK:

- Simulation results were plotted and compared with analytical calculations.
- Deviations between methods were analyzed and discussed in terms of:
    - Numerical precision
    - Simulation timestep
    - Possible modeling simplifications

RESULT

SCARA 3DoF ROBOT



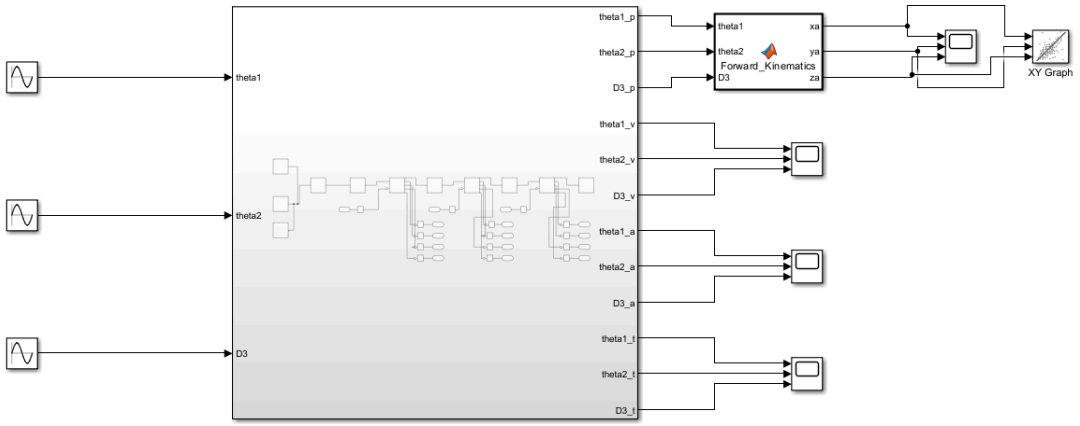
SCARA built in SolidWork

DH Parameters

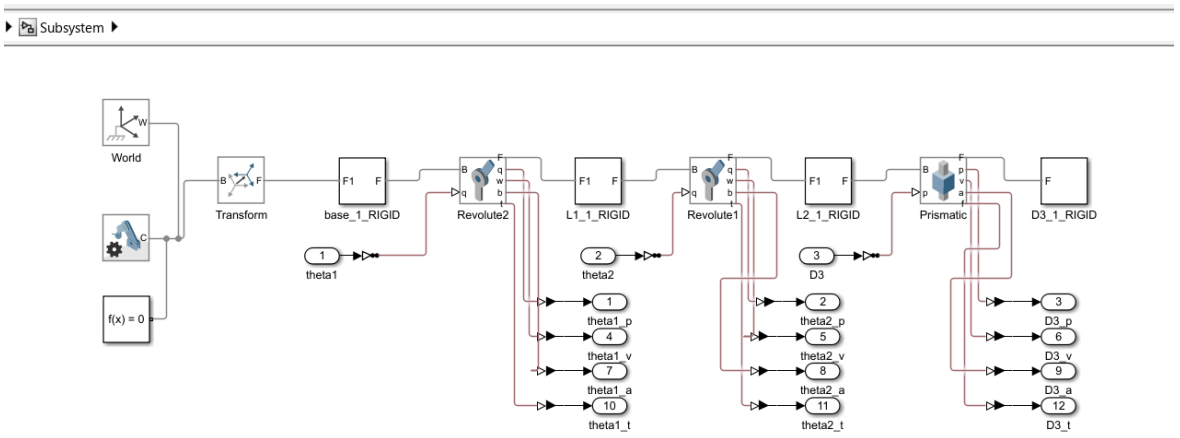
Joint	a (cm)	α (cm)	d (cm)	θ (cm)
0 - 1	80	0	100	θ1
1 - 2	60	0	0	θ2
2 - 3	0	0	-D3	0

Forward Kinematics

Simulation



Simulink circuit



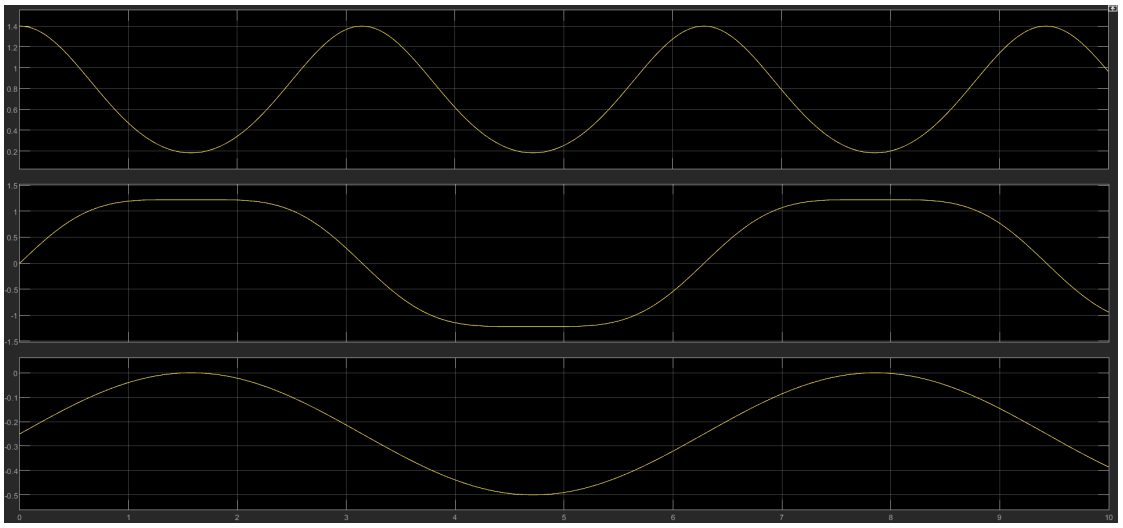
Subsystem

MATLAB Function

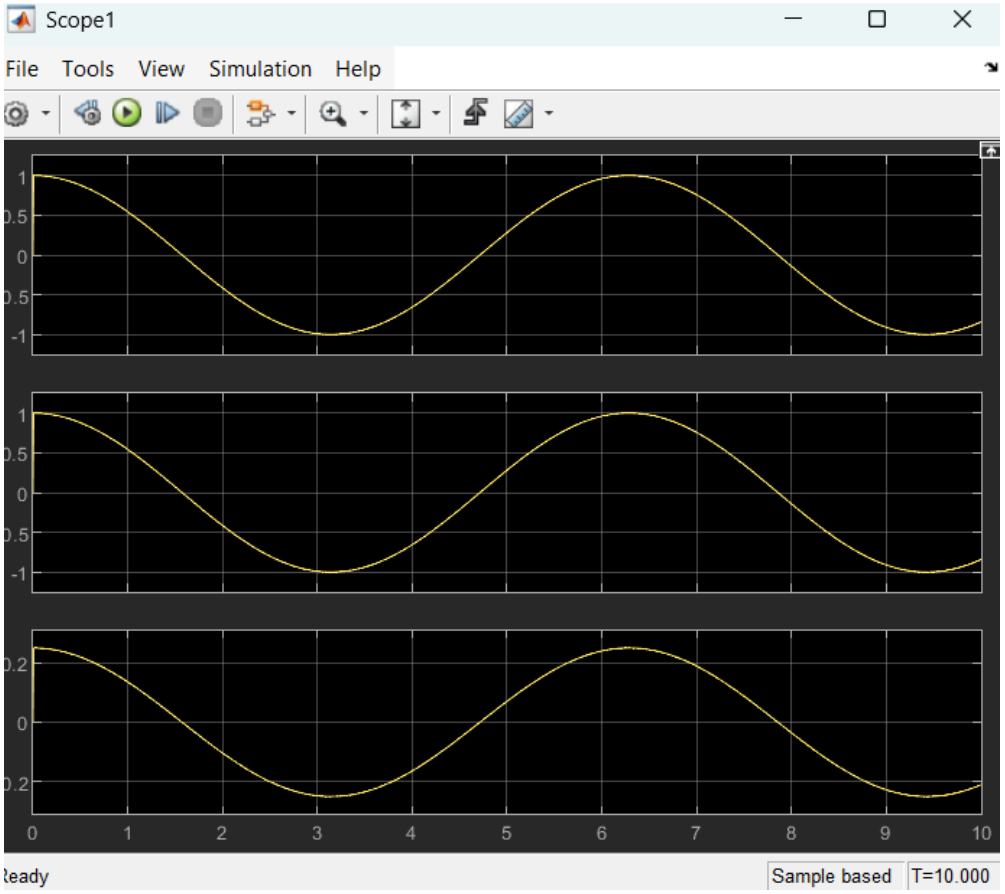
FORWARDKINEMATICS ▸ MATLAB Function

```
1  function [xa, ya, za] = Forward_Kinematics(theta1, theta2, D3)
2  l1=0.8;
3  l2=0.6;
4
5  xa=l1*cos(theta1)+l2*cos(theta1+theta2);
6  ya=l1*sin(theta1)+l2*sin(theta1+theta2);
7  za=D3;
8
```

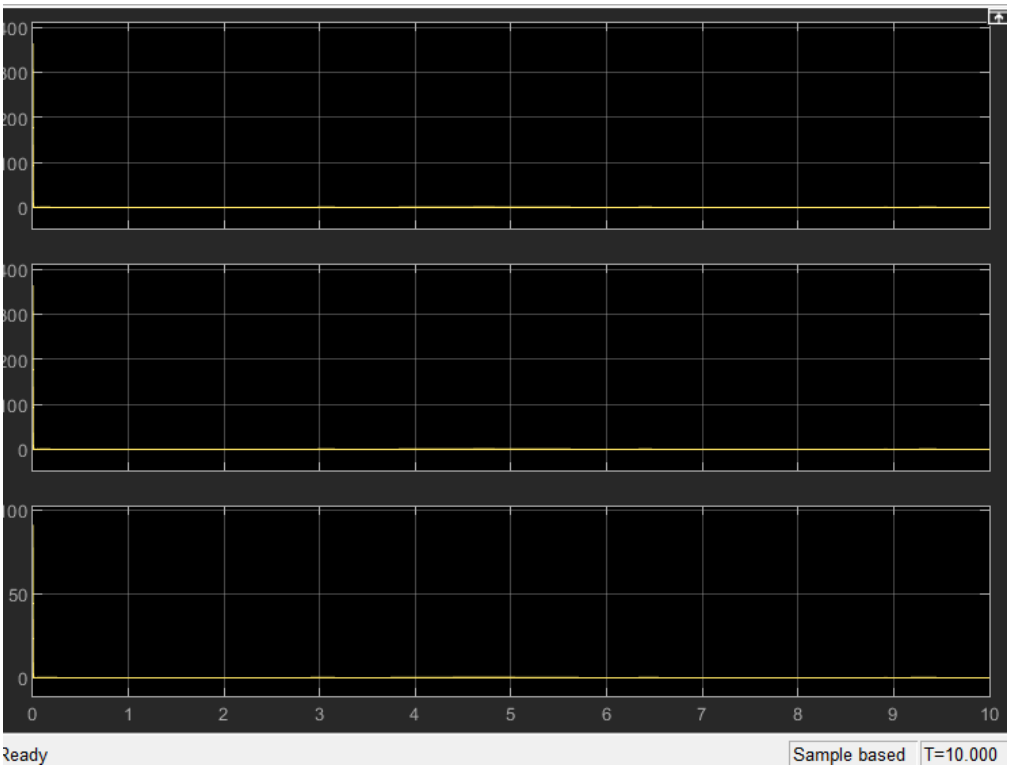
Forward Kinematic Function



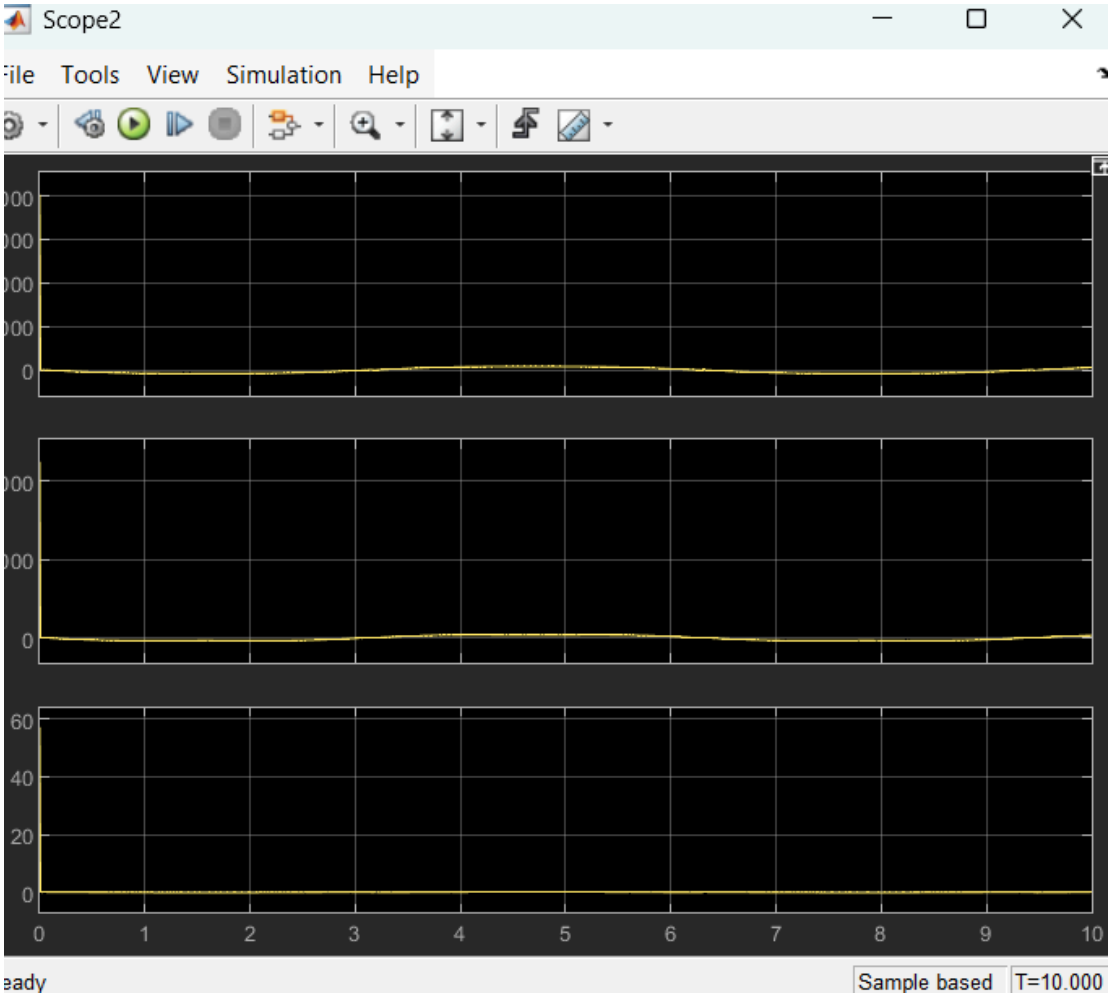
Scope Reading for Position



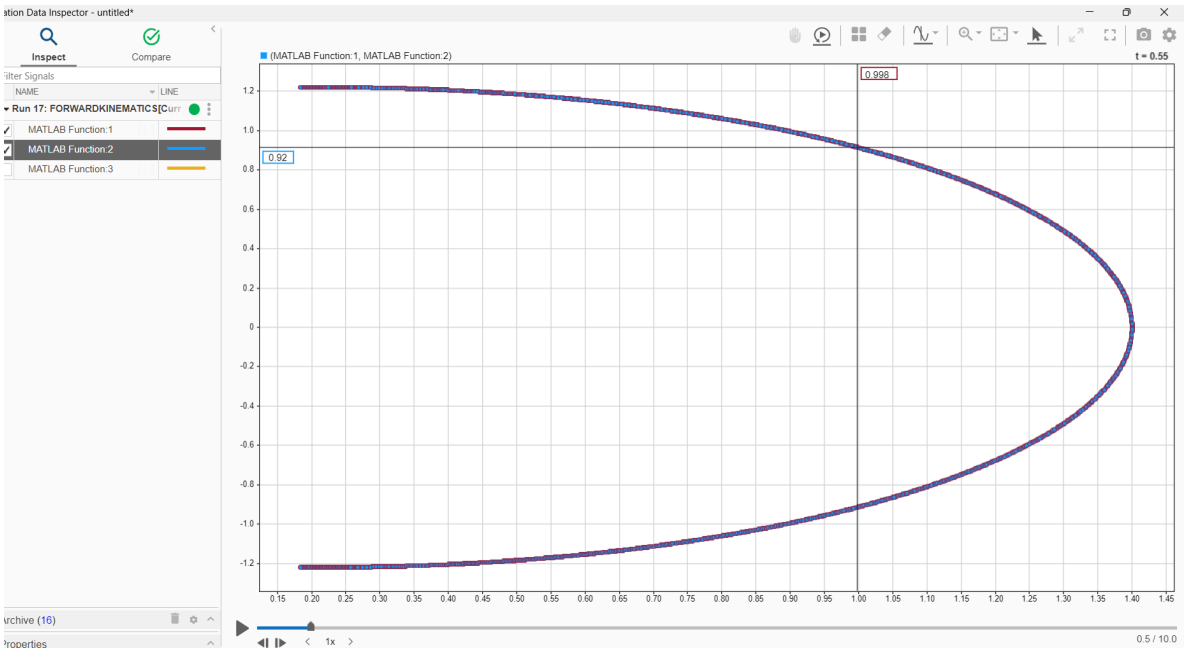
Scope Reading for Velocity



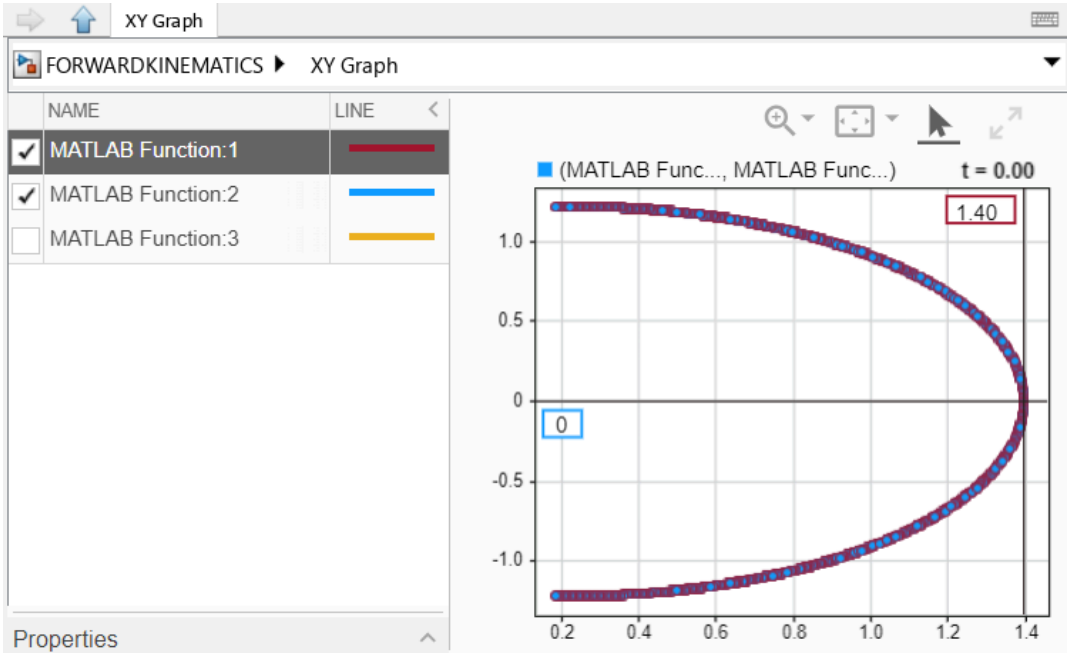
Scope Reading for Acceleration



Scope Reading for Torque



“Simulation Data Inspector” Graph at θ1 = 30 degree and θ2 = 30 degree



XY-Axis Graph at 1 = 0 degree and θ2 = 0 degree

Analytical

MATLAB

Editor - ANALYTICAL.m

Mechanics Explorers - Mechanics Explore

ANALYTICAL.m

```
clc; clear; close all;
syms theta1 theta2 D3

d1=1.00; %in m
a1=0.80;
a2=0.60;

DH=@( a, alpha, d, theta)...
[ cos(theta), -sin(theta)*cos(alpha),  sin(theta)*sin(alpha), a*cos(theta);
  sin(theta),  cos(theta)*cos(alpha), -cos(theta)*sin(alpha), a*sin(theta);
           0,             sin(alpha),             cos(alpha),            d;
           0,                      0,                      0,            1];

A1=DH( a1, 0, d1, theta1)
A2=DH( a2, 0, 0, theta2)
A3=DH( 0, 0, -D3, 0)

T = simplify(A1 * A2 * A3)

theta1_val = deg2rad(30)
theta2_val = deg2rad(30)
D3_val = 0.25;

newT = vpa(subs(T, [theta1 theta2 D3], ...
                   [theta1_val theta2_val D3_val]), 4)
```

Coding for forward kinematics analytically

Workspace | Command Window

```
A1 =

[cos(theta1), -sin(theta1), 0, (4*cos(theta1))/5]
[sin(theta1),  cos(theta1), 0, (4*sin(theta1))/5]
[           0,            0, 1,                1]
[           0,            0, 0,                1]


A2 =

[cos(theta2), -sin(theta2), 0, (3*cos(theta2))/5]
[sin(theta2),  cos(theta2), 0, (3*sin(theta2))/5]
[           0,            0, 1,                0]
[           0,            0, 0,                1]


A3 =

[1, 0, 0,   0]
[0, 1, 0,   0]
[0, 0, 1, -D3]
[0, 0, 0,   1]


T =

[cos(theta1 + theta2), -sin(theta1 + theta2), 0, (3*cos(theta1 + theta2))/5 + (4*cos(theta1))/5]
[sin(theta1 + theta2),  cos(theta1 + theta2), 0, (3*sin(theta1 + theta2))/5 + (4*sin(theta1))/5]
[                   0,                     0, 1,                                       1 - D3]
[                   0,                     0, 0,                                           1]
```

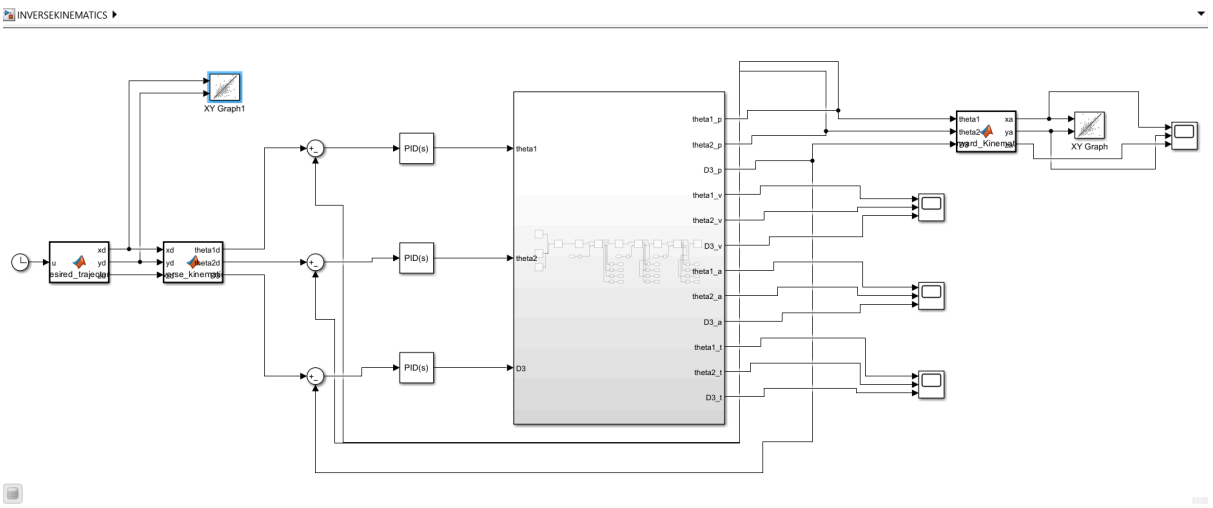
```
theta1_val =

    0.5236


theta2_val =

    0.5236


newT =

[  0.5, -0.866,   0, 0.9928]
[0.866,    0.5,   0, 0.9196]
[    0,      0, 1.0,   0.75]
[    0,      0,   0,    1.0]

fx >>
```

The analytical at θ1 at = 30 degree and θ2 at 30 degree

```
theta1_val =

     0


theta2_val =

     0


newT =

[1.0,   0,   0,  1.4]
[  0, 1.0,   0,    0]
[  0,   0, 1.0, 0.75]
[  0,   0,   0,  1.0]
```

The analytical at θ1 at = 0 degree and θ2 at 0 degree

Inverse Kinematics Simulation

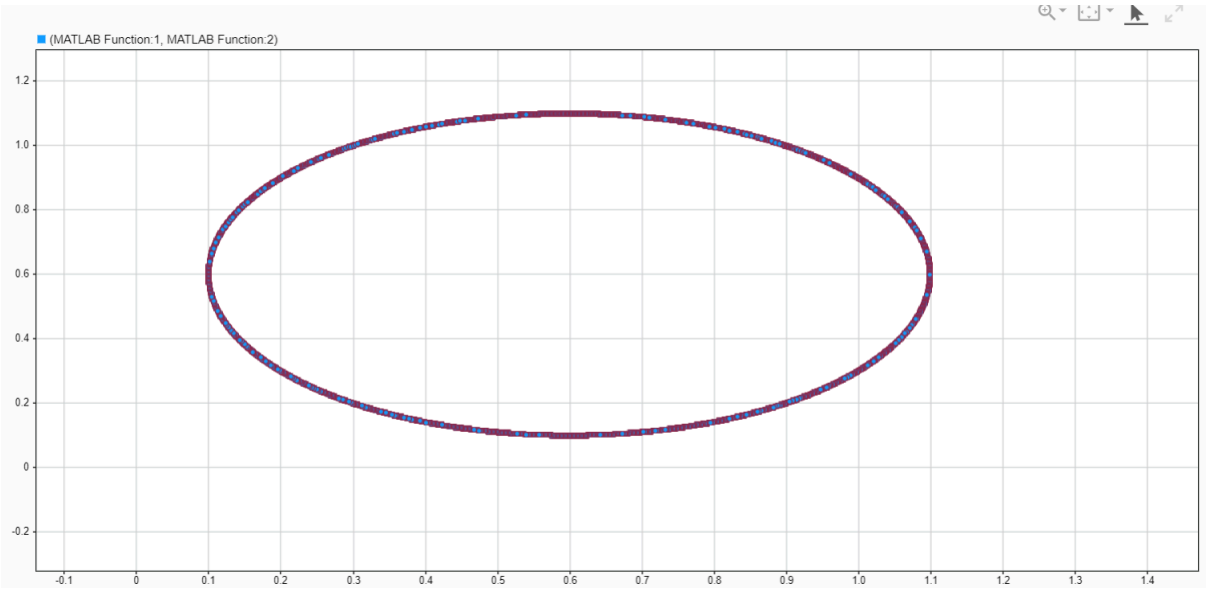


The Inverse Kinematic Simulink Circuit

XY Graph1 | MATLAB Function1

INVERSEKINEMATICS ▸ MATLAB Function1

```
1  function [theta1d,theta2d,D3] = inverse_kinematics(xd,yd,zd)
2
3  l1=0.8;
4  l2=0.6;
5  D3=-0.5;
6
7  v=(xd^2+yd^2-l1^2-l2^2)/(2*l1*l2);
8  theta2d=acos(v);
9  theta1d=atan(yd/xd)-atan((l2*sin(theta2d))/(l1+l2*cos(theta2d)));
10
```

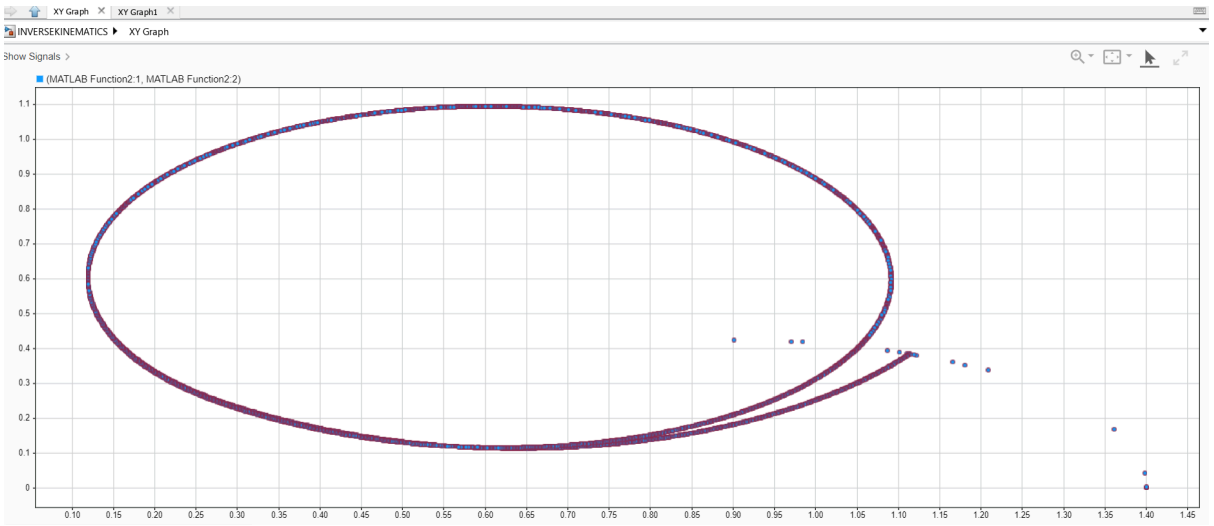


The desired trajectory coding and graph

MATLAB Function1 | XY Graph1

INVERSEKINEMATICS ▸ MATLAB Function1

```
1  function [theta1d,theta2d,D3] = inverse_kinematics(xd,yd,zd)
2
3  l1=0.8;
4  l2=0.6;
5  D3=-0.5;
6
7  v=(xd^2+yd^2-l1^2-l2^2)/(2*l1*l2);
8  theta2d=acos(v);
9  theta1d=atan(yd/xd)-atan((l2*sin(theta2d))/(l1+l2*cos(theta2d)));
10
```



The inverse kinematics coding and result

DISCUSSION

The analysis of the SCARA robot was carried out through both analytical calculations and simulation, focusing on forward and inverse kinematics performance. For the forward kinematics (FK), a comparison was made between the simulated output from Simulink and the theoretical values obtained through analytical methods.The position equation for this simulator is :

$$x(t) = (3 * cos(theta1 + theta2))/5 + (4 * cos(theta1))/5$$

$$y(t) = (3 * sin(theta1 + theta2))/5 + (4 * sin(theta1))/5$$

$$z(t) = 1 - D3$$

When joint angles of $\theta_1 = 30°$ and $\theta_2 = 30°$, the robot produced an end-effector X position at 0.998 m and a Y position at 0.920 m and the analytical calculation gave X = 0.9928 m and Y = 0.9196 m. The error in the X position is approximately 0.0002 m, or 0.02%, while the Y-position error is 0.0732 m, 7.3%. These small differences are within acceptable limits because of numerical solver tolerances, interpolation of sine inputs, and rounding errors in Simulink's simulation environment. Nonetheless, the close match between the two sets of values validates the correctness of the DH parameter configuration and forward kinematics expressions used in both approaches.

In the inverse kinematics (IK) section, a trajectory-following control system was implemented using a feedback loop with PID controllers. PID controller is used to improve the end effector path to get a better value where it is more accurate and precise. The desired end-effector path was generated via a MATLAB function and then converted into joint targets through an inverse kinematics function block. These target joint values were compared with the actual joint positions obtained from the Simscape model, and the resulting error was sent into a PID controller. The controller is set with proportional gain P=10 and integral gain I=20, while the derivative gain was set to zero. This tuning allowed the robot to follow the target trajectory smoothly while minimizing steady-state error. The proportional term helped with quick error correction, while the integral term eliminated residual offset over time.

During simulation, the robot successfully tracked the desired motion path. The joint positions responded quickly to changes in target values, and the overall system remained stable. There were slight overshoots at the beginning of the motion, likely due to the high integral gain and quickly settled also followed with high accuracy trajectory. Any small deviation was attributed to the system and time required for the controller to compensate.

Overall, both the forward and inverse kinematics analyses demonstrated strong alignment between the theoretical and simulated models. The errors observed were minimal and expected due to the nature of real-time simulation. The PID controller settings used in this project proved effective for basic trajectory tracking, and further improvements could be made with the inclusion of a derivative term or advanced control methods. This study highlights the value of using both analytical and simulation-based approaches to validate robotic system performance in design and control tasks.

CONCLUSION

In conclusion, this project successfully analysed the kinematic behavior of a 3-DoF SCARA robot using both analytical and simulation approaches. The forward kinematics model was verified through comparison between analytical equations and Simulink simulation outputs also the only minor differences can be observed which showed errors of less than 1%, which confirms the accuracy of the mathematical model and Simscape implementation. In the inverse kinematics analysis, with PID controller was implemented to the robot, its performed well and allowed the robot to track the target path with acceptable

accuracy and system stability. Overall, the combination of analytical calculations and simulation tools increase the understanding of the robot's motion and control performance.