

PRESENTATION on Learnings from Tasks

INDEX

- Stage 1 Transformations

DOF, RoboAnalyzer, Forward Kinematics, Inverse Kinematics, D-H parameters

- Stage 2 Kinematics

MATLAB program of Forward Kinematics Inverse Kinematics for 2 & 3 planar manipulator

- Stage 3 Motion Planning

- Stage 4 Final Execution

Nitish Kumar (Team Coordinator)

Abhishek Shimpi (Member)

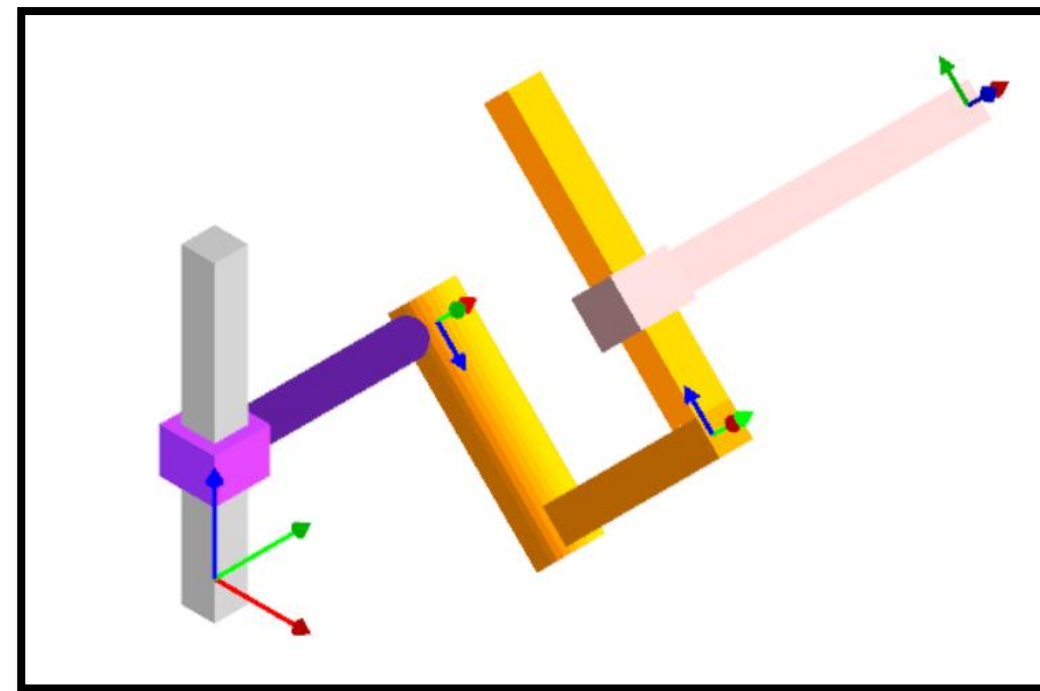
RoboAnalyzer Online Competition 2021

Team A3

Forward Kinematics

In forward kinematics for positions, the joint positions, i.e., the angles of revolute joints and the displacements of prismatic joints, are prescribed. The task is to find the end-effector's configuration or pose, i.e., its position and orientation.

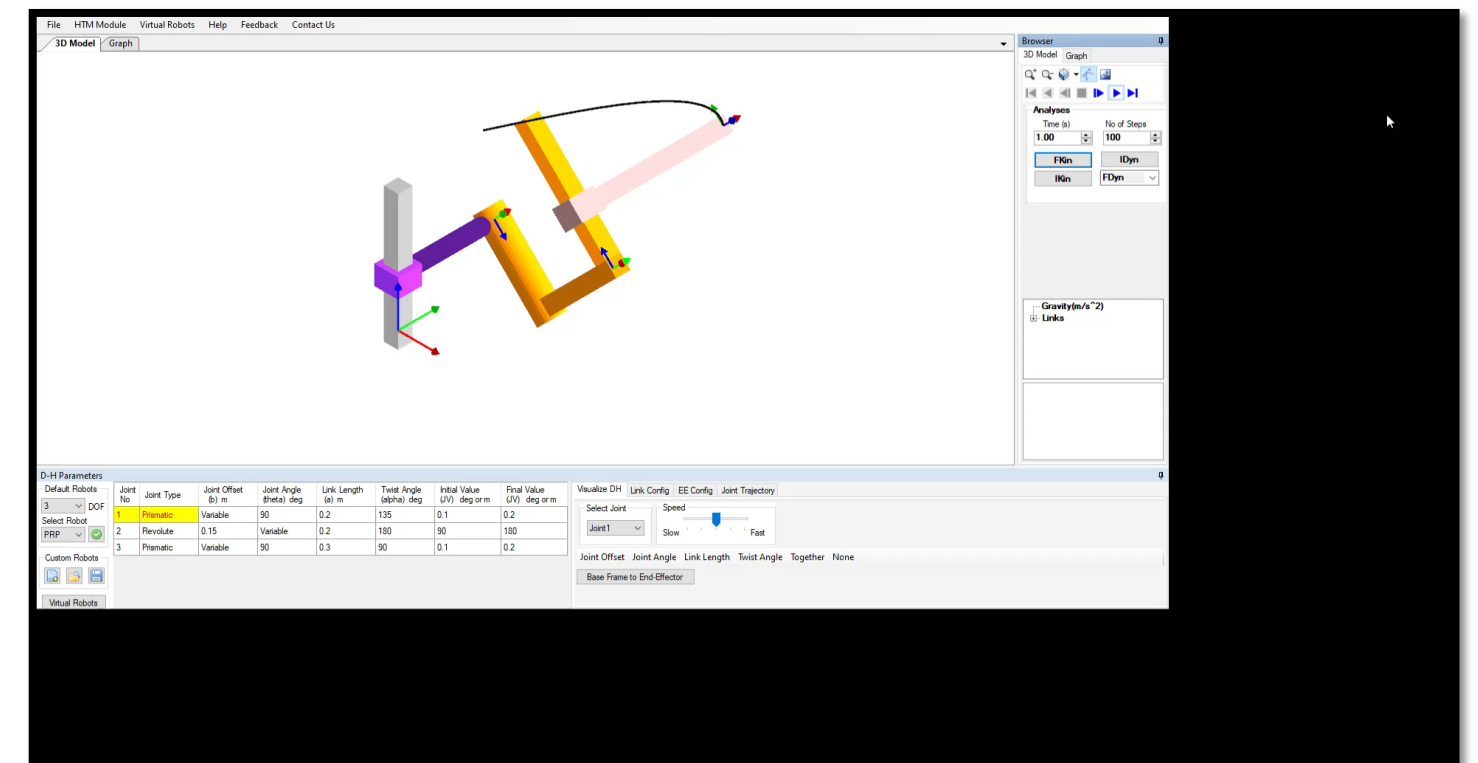
We define D-H Parameters.



3 DOF with PRP configuration

Four D-H Parameter

1. Joint offset (b) m
2. Joint Angle (θ) deg
3. Link length (a) m
4. Twist angle (α) deg

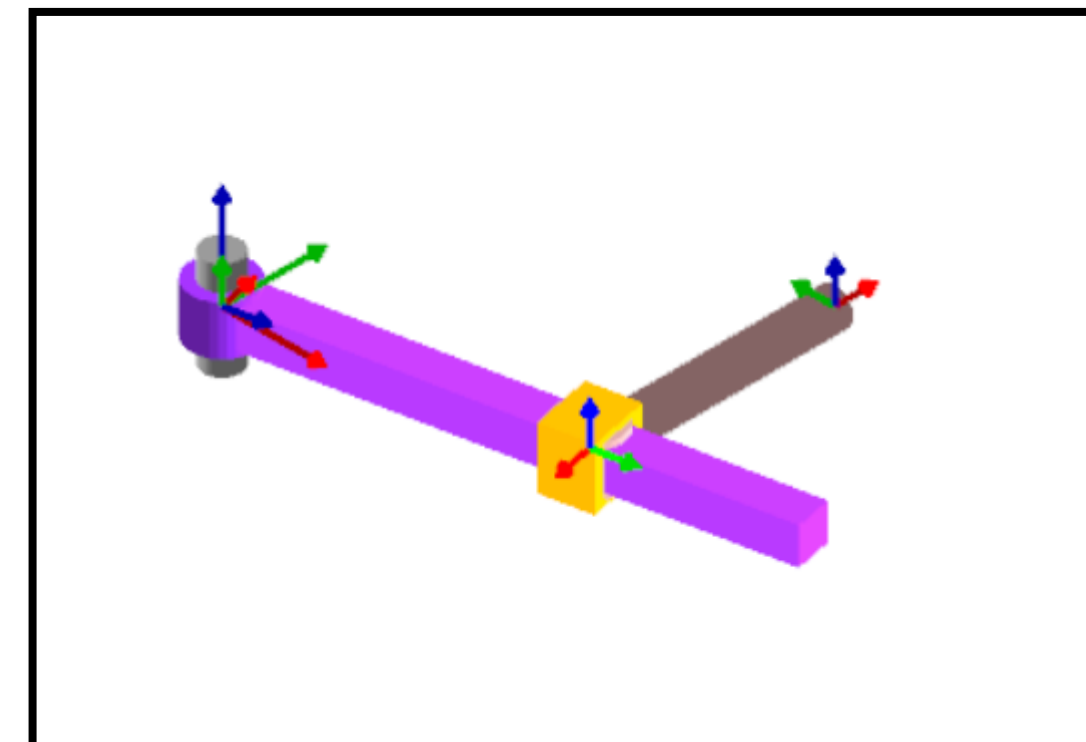
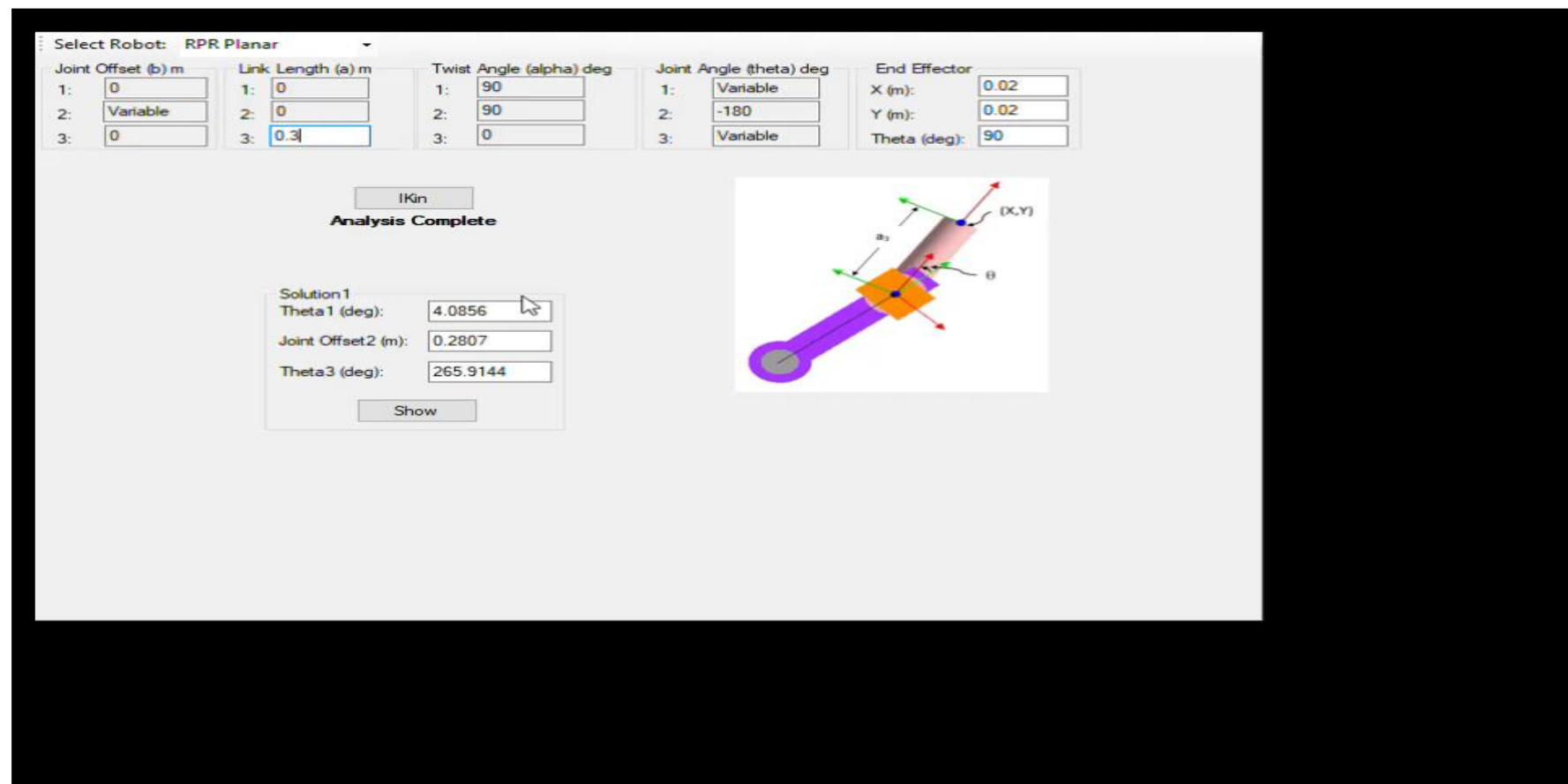


Motion video of 3DOF PRP

D-H Parameters								
Joint No	Joint Type	Joint Offset (b) m	Joint Angle (theta) deg	Link Length (a) m	Twist Angle (alpha) deg	Initial Value (JV) deg or m	Final Value (JV) deg or m	
1	Prismatic	Variable	90	0.2	135	0.1	0.2	
2	Revolute	0.15	Variable	0.2	180	90	180	
3	Prismatic	Variable	90	0.3	90	0.1	0.2	

Inverse Kinematics

The inverse kinematics problem consists of the determination of the joint variables corresponding to a given end-effector's orientation and position

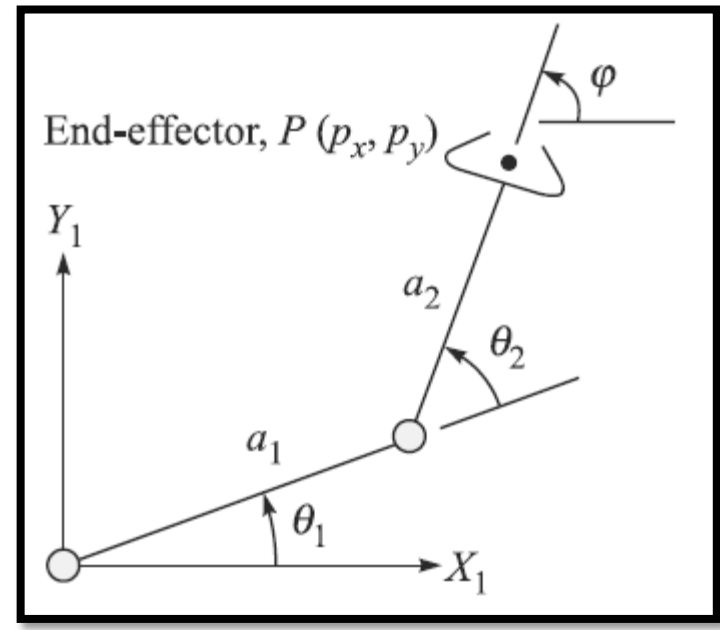


Output of Inverse Kinematics

Forward Kinematics of a Two-link Planar Arm

$\cos\theta$	$-\sin\theta$	0	P_x
$\sin\theta$	$\cos\theta$	0	P_y
0	0	1	0
0	0	0	1

$P_x = a_1 \cdot \cos\theta_1 + a_2 \cdot \cos\theta_{12}$
 $P_y = a_1 \cdot \sin\theta_1 + a_2 \cdot \sin\theta_{12}$
 $\theta_{12} = \theta_1 + \theta_2 - \theta_i$



Inverse Kinematics of a Two-link Planar Arm

I. Algebraic solution: Equating elements (2,1), (1,1), (1,4), and (2,4) of the two matrices, we get:

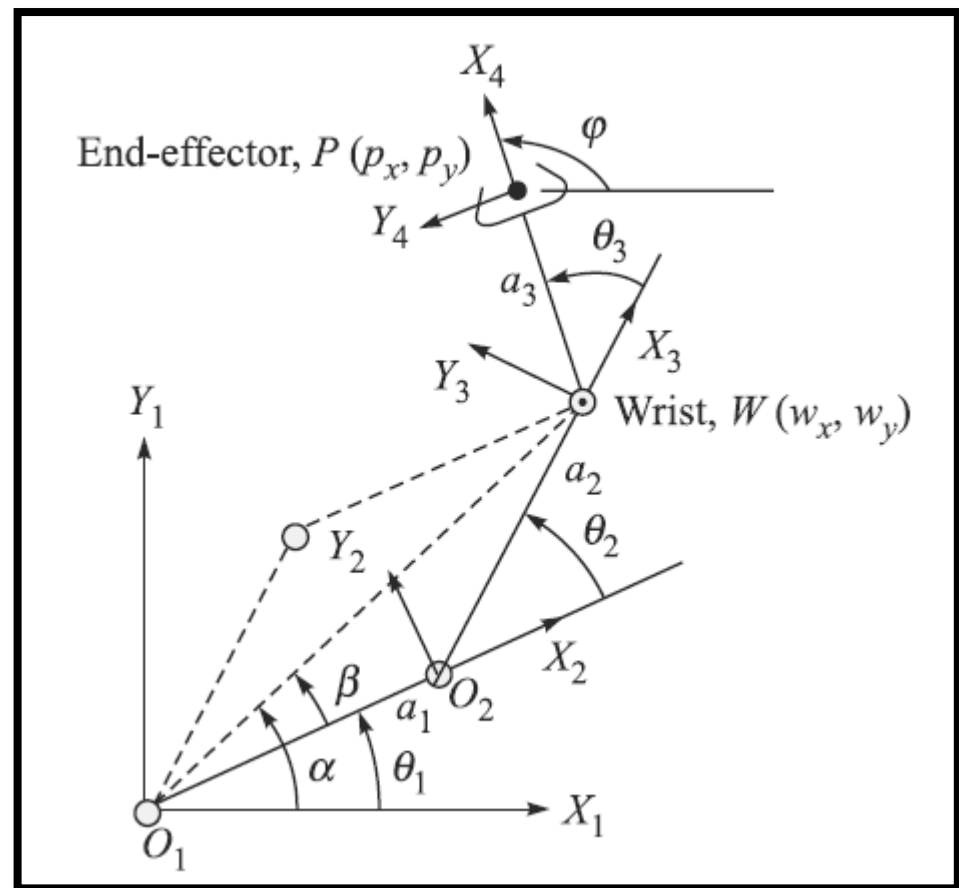
$S_{12} = n_y$ and $C_{12} = n_x \rightarrow \theta_{12} = ATAN2(n_y, n_x)$
 $a_2 C_{12} + a_1 C_1 = p_x$ or $a_2 n_x + a_1 C_1 = p_x \rightarrow C_1 = \frac{p_x - a_2 n_x}{a_1}$
 $a_2 S_{12} + a_1 S_1 = p_y$ or $a_2 n_y + a_1 S_1 = p_y \rightarrow S_1 = \frac{p_y - a_2 n_y}{a_1}$
 $\theta_1 = ATAN2(S_1, C_1) = ATAN2\left(\frac{p_y - a_2 n_y}{a_1}, \frac{p_x - a_2 n_x}{a_1}\right)$

Since θ_1 and θ_{12} are known, θ_2 can also be calculated.

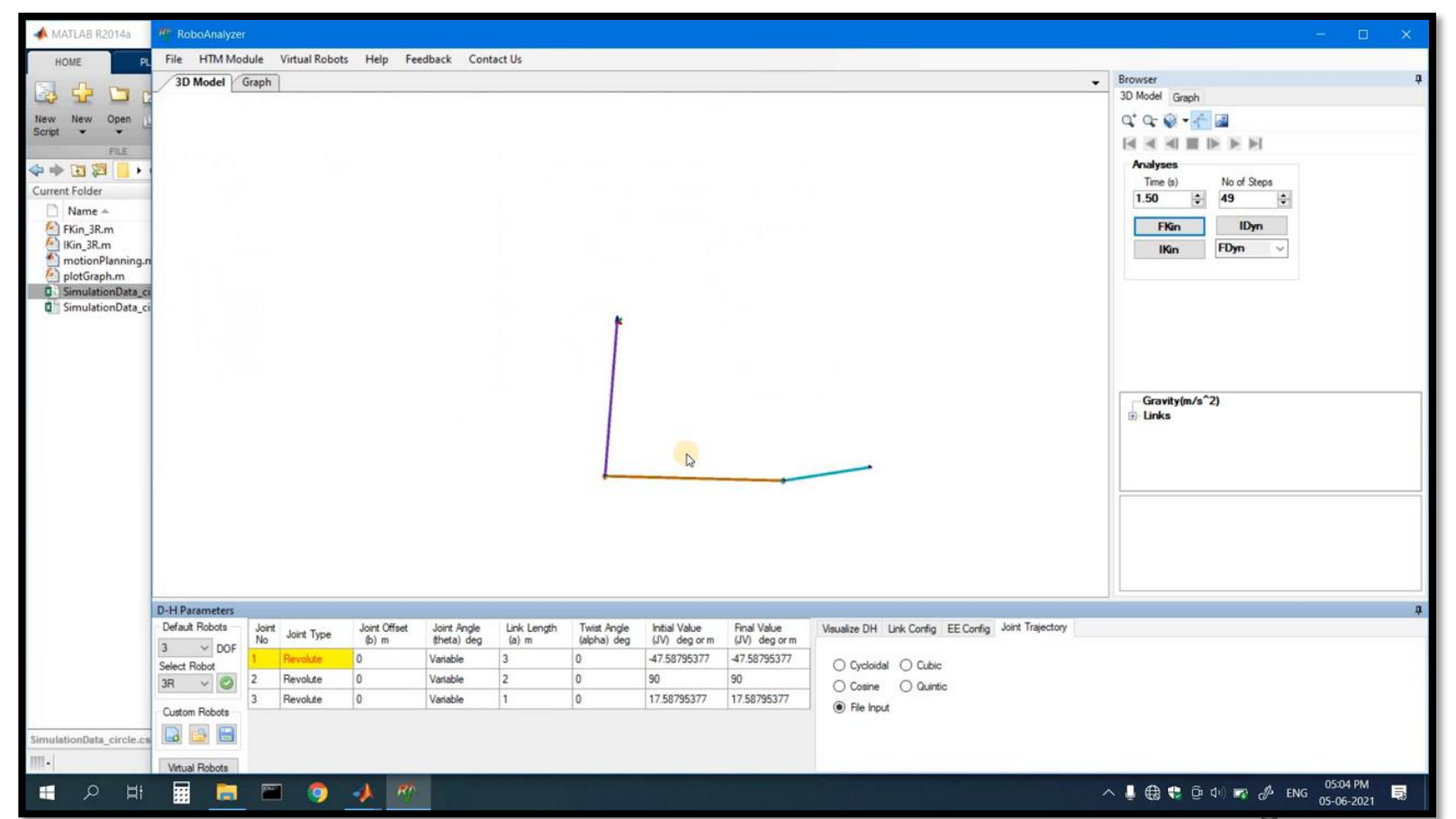
Forward Kinematics of a Three-link Planar Arm

$\cos\theta$	$-\sin\theta$	0	P_x
$\sin\theta$	$\cos\theta$	0	P_y
0	0	1	0
0	0	0	1

$P_x = a_1 \cdot \cos\theta_1 + a_2 \cdot \cos\theta_{12} + a_3 \cdot \cos\theta_{123}$
 $P_y = a_1 \cdot \sin\theta_1 + a_2 \cdot \sin\theta_{12} + a_3 \cdot \sin\theta_{123}$
 $\theta_{12} = \theta_1 + \theta_2 - \theta_i$
 $\theta_{123} = \theta_{12} + \theta_3$



Motion Planning of Three-link Arm to form a Circle



Stage 3 Motion Planning

With the help of MATLAB we create an csv file which has geometry file which can be drawn through robot in RoboAnalyzer Software.

This this the variable value which will be exported to csv to be use in Roboanlyser

X, Y and Z Coordinates

End effector angles(does not changed in current program)

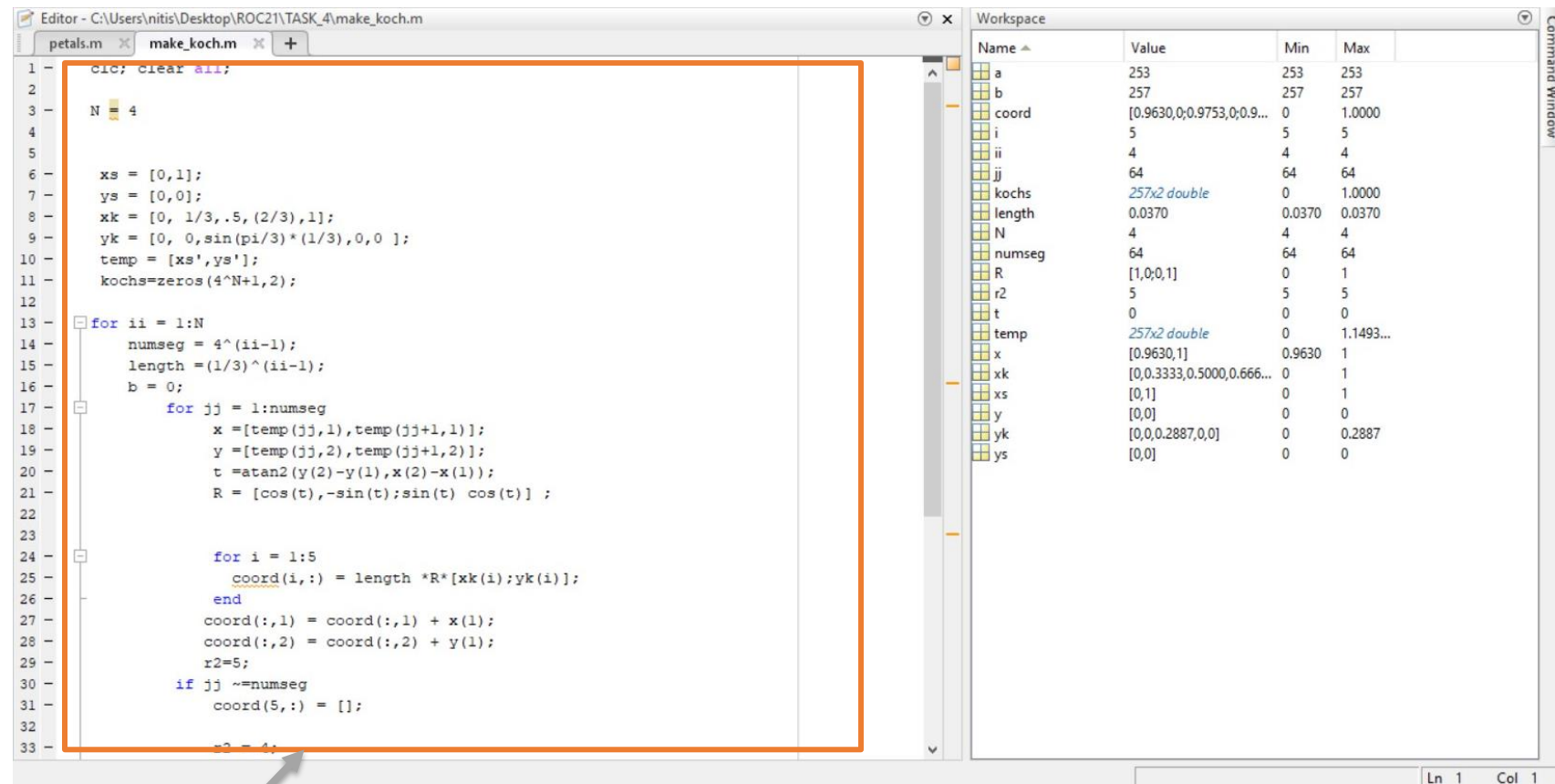
The MATLAB workspace shows the following variables:

Name	Value	Min	Max
centerY	191.5111	191.51...	191.51...
centerZ	191.5111	191.51...	191.51...
dbugVX	300x1 double	800	800
dbugVY	300x1 double	-160.6...	160.69...
dbugVZ	300x1 double	1005	1.3572...
HfinalYPos	191.5111	191.51...	191.51...
HfinalZPos	-160.6969	-160.6...	-160.6...
HinitYPos	0	0	0
HinitZPos	0	0	0
HpYarray	1x300 double	0	352.18...
HpZarray	1x300 double	-160.6...	160.69...
i	300	300	300
lenBud	250	250	250
numIter	100	100	100
numPts	300	300	300
offset	200	200	200
pXarray	1200x1 double	800	800
pYarray	1200x1 double	-352.1...	352.18...
pZarray	1200x1 double	652.81...	1.3572...
tArray	1x100 double	0	1

	A	B	C	D	E	F	G	H	I	J	K
1	800	0	1005	90	0	90					
2	800	1.934456	1003.377	90	0	90					
3	800	3.868911	1001.754	90	0	90					
4	800	5.803367	1000.13	90	0	90					
5	800	7.737823	998.5072	90	0	90					
6	800	9.672278	996.884	90	0	90					
7	800	11.60673	995.2608	90	0	90					
8	800	13.54119	993.6376	90	0	90					
9	800	15.47565	992.0144	90	0	90					
10	800	17.4101	990.3912	90	0	90					
11	800	19.34456	988.768	90	0	90					
12	800	21.27901	987.1448	90	0	90					
13	800	23.21347	985.5216	90	0	90					
14	800	25.14792	983.8984	90	0	90					
15	800	27.08238	982.2752	90	0	90					
16	800	29.01683	980.652	90	0	90					
17	800	30.95129	979.0288	90	0	90					
18	800	32.88575	977.4056	90	0	90					

Stage 4 Final Execution

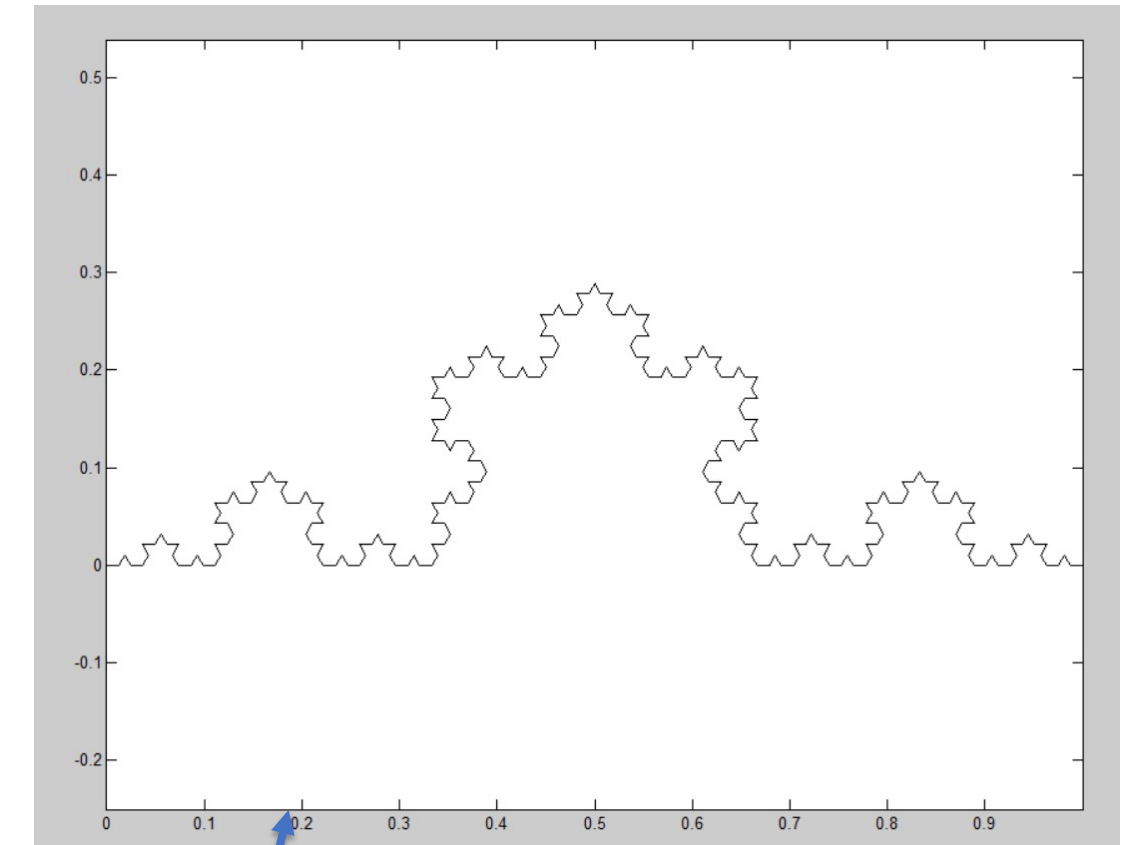
In Final submission we executed the motion Planning again for Artistic Design with the help of MATLAB & csv file which we imported to Virtual Robot Manipulator. We are creating a fractal pattern Koch curve from matlab and importing the csv file containing the x, y and z coordinates of curve to Roboanalyser.



The screenshot shows the MATLAB Editor with a script named 'make_koch.m'. The code defines a function to generate a Koch curve. It starts with clearing all variables and setting N=4. It then defines the initial segment coordinates (xs, ys) and the rotation angle (pi/3). The main loop iterates over the number of segments (N), calculating the number of segments (numseg), the length of each segment (length), and the rotation angle (t). It then calculates the coordinates of the curve segments (coord) and the rotation matrix (R). The workspace shows the following variables:

Name	Value	Min	Max
a	253	253	253
b	257	257	257
coord	[0.9630,0,0.9753,0,0.9...	0	1.0000
i	5	5	5
ii	4	4	4
jj	64	64	64
kochs	257x2 double	0	1.0000
length	0.0370	0.0370	0.0370
N	4	4	4
numseg	64	64	64
R	[1,0,1]	0	1
r2	5	5	5
t	0	0	0
temp	257x2 double	0	1.1493...
x	[0.9630,1]	0.9630	1
xk	[0,0.3333,0.5000,0.666...	0	1
xs	[0,1]	0	1
y	[0,0]	0	0
yk	[0,0,0.2887,0,0]	0	0.2887
ys	[0,0]	0	0

Matlab program to create Koch curve



Fractal pattern created by matlab program. This will be then scaled and exported to csv file and then will be used in Roboanalyser.

Generating Koch curve using csv file from the matlab program



THANK YOU