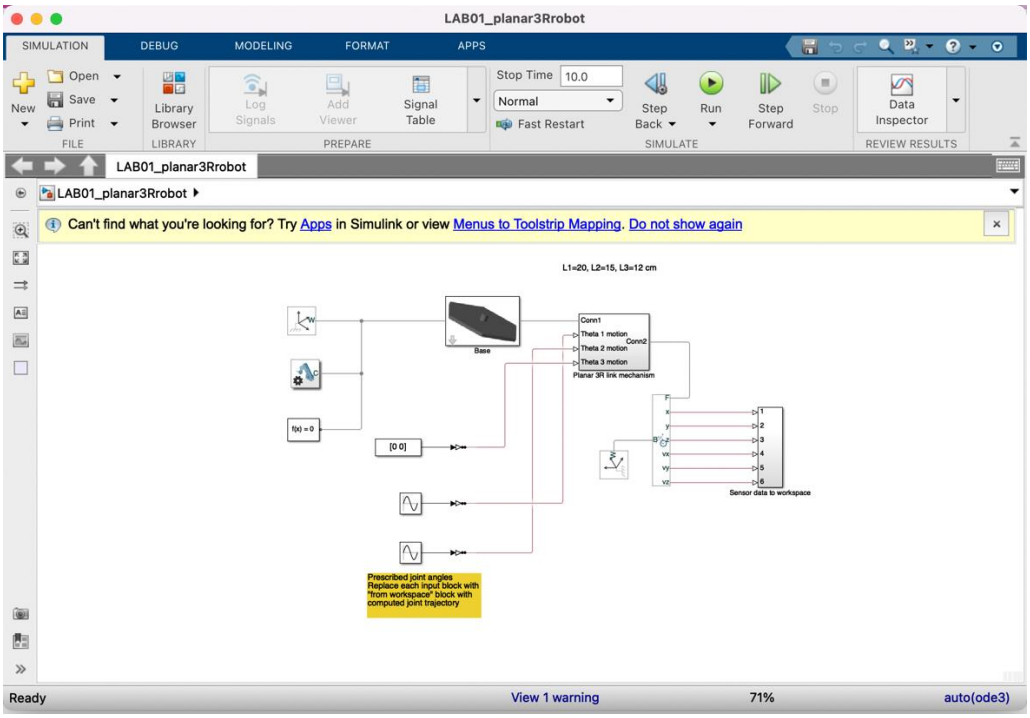
INVERSE KINEMATIC AND TRAJECTORY PLANNING

MEM675
Spring 2023
Euisun Kim

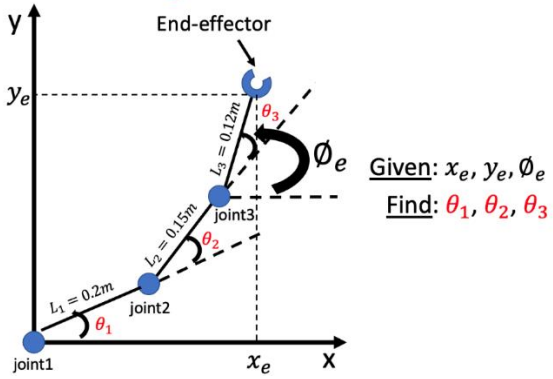
Part A: Create Inverse Kinematic Algorithm of 3R manipulator

A.1 – Planar 3R robot model

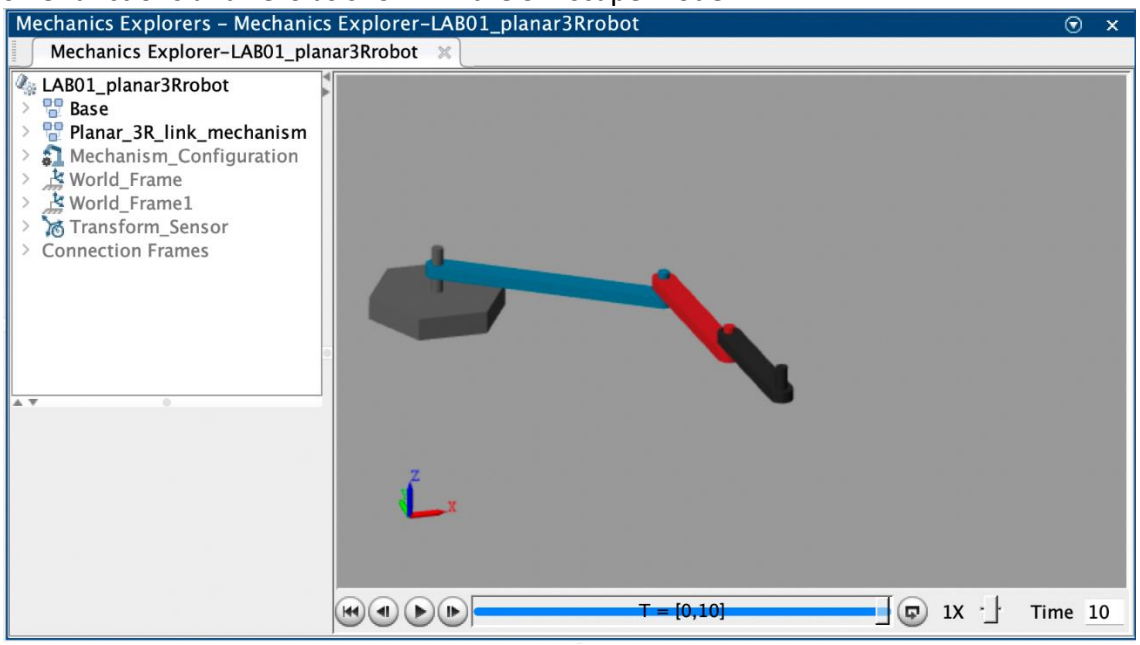
- Open 'Lab1_planar3Rrobot.slx' file to open Simscape Multibody.
- Click 'Run' on the 'Simulation' tab to start the simulation



- Planar 3R robot below will show. This planar robot has three revolute joints that connect three links. Each link has length following; $L_1 = 0.2, L_2 = 0.15, and\ L_3 = 0.12$

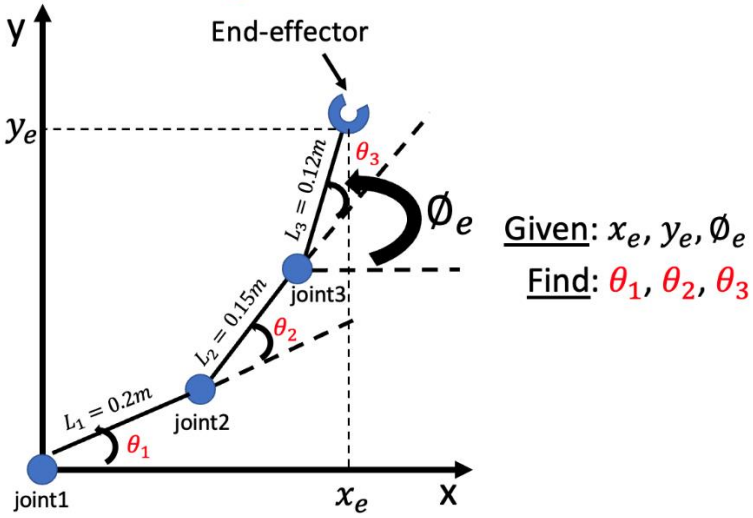


- This planar 3R robot will show some motion during the simulation time following the angle values given to ‘Theta1 motion‘, ‘Theta2 motion’, and ‘Theta3 motion’ which are sine functions and zero as shown in the Simscape model.



A.2 – Analytic Solution for 3R inverse kinematics

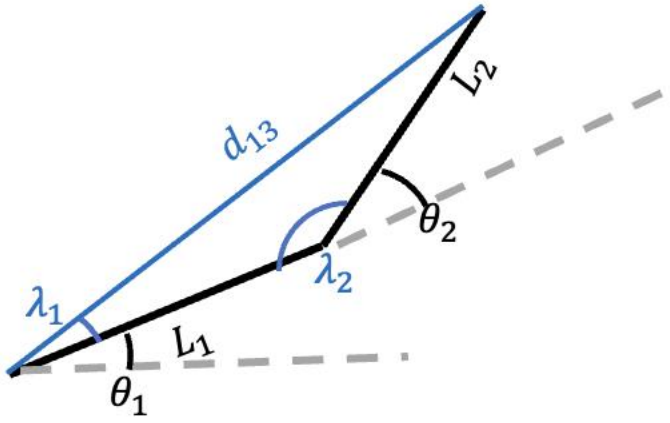
Now we will derive analytical solution for inverse kinematic of this 3R arm.
Follow steps below to derive the solution.



Step 1) Get joint 3 location (x_3, y_3)

$$x_3 = x_e - L_3 * \cos(\emptyset_e)$$
$$y_3 = y_e - L_3 * \sin(\emptyset_e)$$

Step 2) Find θ_2 using cosine laws from the triangle below that consists of joint1, joint2 and joint3.



$$d_{13}^2 = L_1^2 + L_2^2 - 2L_1L_2\cos\gamma_2$$
$$\cos\gamma_2 = \frac{L_1^2 + L_2^2 - d_{13}^2}{2L_1L_2}$$
$$\cos\theta_2 = \cos(\pi - \gamma_2) = -\cos\gamma_2 = \frac{d_{13}^2 - L_1^2 - L_2^2}{2L_1L_2}$$
$$\theta_2 = \pm atan2(\sqrt{1-(\cos\theta_2)^2}, \cos\theta_2)$$

Step 3) Find θ_1.

*** We will use a different approach than we used in the lecture to find out the solution that universally applied regardless of the robot configuration.*

Joint 3 location (x_3, y_3) can be represented by following equations,

$$x_3 = L_1\cos\theta_1 + L_2\cos(\theta_1 + \theta_2)$$
$$y_3 = L_1\sin\theta_1 + L_2\sin(\theta_1 + \theta_2)$$

Using the properties of trigonometric functions,

$$\sin(A+B) = \sin A\cos B + \cos A\sin B$$
$$\cos(A+B) = \cos A\cos B - \sin A\sin B$$

Joint 3 location (x_3, y_3) can be rearranged like this.

$$x_3 = L_1\cos\theta_1 + L_2\cos(\theta_1 + \theta_2) = L_1\cos\theta_1 + L_2(\cos\theta_1\cos\theta_2 - \sin\theta_1\sin\theta_2)$$
$$y_3 = L_1\sin\theta_1 + L_2\sin(\theta_1 + \theta_3) = L_1\sin\theta_1 + L_2(\sin\theta_1\cos\theta_2 + \cos\theta_1\sin\theta_2)$$

In a matrix form,

$$\begin{bmatrix} x_3 \\ y_3 \end{bmatrix} = \begin{bmatrix} L_1 + L_2\cos\theta_2 & -L_2\sin\theta_2 \\ L_2\sin\theta_2 & L_1 + L_2\cos\theta_2 \end{bmatrix} \begin{bmatrix} \cos\theta_1 \\ \sin\theta_1 \end{bmatrix}$$

$$\begin{bmatrix} \cos\theta_1 \\ \sin\theta_1 \end{bmatrix} = \begin{bmatrix} L_1 + L_2\cos\theta_2 & -L_2\sin\theta_2 \\ L_2\sin\theta_2 & L_1 + L_2\cos\theta_2 \end{bmatrix}^{-1} \begin{bmatrix} x_3 \\ y_3 \end{bmatrix}$$

$$\theta_1 = atan2(\sin\theta_1, \cos\theta_1)$$

Step 4) Find θ_3

$$\emptyset_e = \theta_1 + \theta_2 + \theta_3$$
$$\theta_3 = \emptyset_e - \theta_1 - \theta_2$$

A.3 – Develop Inverse Kinematic Algorithm

In this section, we will develop inverse kinematic algorithm using the analytical solution we derived above.

- Open 'LAB01_planar3Rrobot_inversekinematics.m' and fill out the blank to develop inverse kinematic algorithm using the derived equations in A.2.
- When developing the algorithm, follow the predefined variable names as written in the MATLAB code comment.

A.4 – Plot Inverse Kinematic Solution to Verify your Inverse Kinematic Alorithm

Use the code below to verify your inverse Kinematic Algorithm is working properly. Add the code below at the end of your 'LAB01_planar3Rrobot_inversekinematics.m'. Run the code and check if the end effector location (red dot) on the figure is the same position(**xe**,**ye**) and orientation(**phie**) as you specified at the beginning of 'LAB01_planar3Rrobot_inversekinematics.m' code.

If the end effector location is different, debug your code!

MATLAB CODE

```
%% plot inverse kinematics solution
theta1=theta1inv
theta2=theta2inv
theta3=theta3inv

S1=sin(theta1);
S12=sin(theta1+theta2);
S2=sin(theta2);
S123=sin(theta1+theta2+theta3);
S23=sin(theta2+theta3);
S3=sin(theta3);
C1=cos(theta1);
```

```
C2=cos(theta2);
C3=cos(theta3);
C12=cos(theta1+theta2);
C23=cos(theta2+theta3);
C123=cos(theta1+theta2+theta3);

%from manipulator geometries
x4pos=a1*C1+a2*C12+a3*C123
y4pos=a1*S1+a2*S12+a3*S123

x3pos=a1*C1+a2*C12;
y3pos=a1*S1+a2*S12;

figure(2)
hold on
plot([0 a1*C1 a1*C1+a2*C12 a1*C1+a2*C12+a3*C123 x4pos],[0 a1*S1
a1*S1+S12*a2 a1*S1+S12*a2+a3*S123 y4pos])
plot(x4pos,y4pos,'o');
axis equal
grid
```

Part B: Create Inverse Kinematic Function

Open `IK.m` matlab function file. You will complete this function `IK.m` so that this function calculates inverse kinematic solutions for 3R manipulator. This function takes end effector position & orientation (x_e, y_e, ϕ_e) and manipulator link lengths (L_1, L_2, L_3) as inputs and results in joint angles $q = [\theta_1, \theta_2, \theta_3]$ as outputs.

```
function q=IK(xe,ye,phie, L1, L2, L3)
% This function outputs inverse kinematic solutions for 3R manipulator
% input : xe,ye,phie, L1, L2, L3
% output : q = [theta1; theta2; theta3]


%% Copy & paste your code from LAB1 here









q=[theta1inv; theta2inv; theta3inv];
end
```

Copy & paste a portion of your LAB01_planar3Rrobot_inversekinematics.m into IK.m so that this function calculates joint angles ($q = [\theta_1, \theta_2, \theta_3]$) from given information ($x_e, y_e, \phi_e, L_1, L_2, L_3$).

% Make sure you are using the same variable names for input and output as written in the matlab code (xe,ye,phie, L1, L2, L3, q)
% For simplicity, choose **elbow minus** solution.

Part C: Create Function that Plots Manipulator's Trajectory

Open plotManipulator.m matlab function file. You will complete this function so that this plots the Manipulator's Trajectory with given joint angles. This function takes joint angles ($q = [\theta_1, \theta_2, \theta_3]$) and manipulator link lengths (L_1, L_2, L_3) as inputs and plots the manipulator's trajectory.

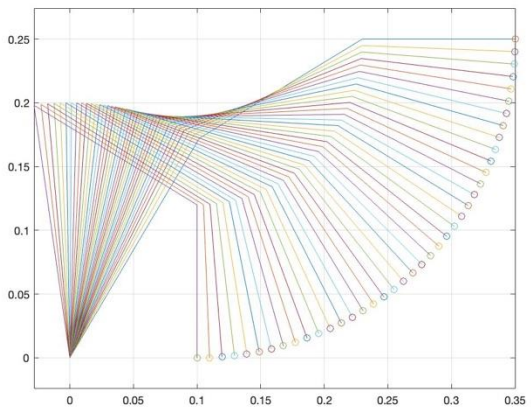
```
function plotManipulator_sol(q, L1, L2, L3)
% This function plots inverse kinematic solutions for 3R manipulator
% input : q, L1, L2, L3



%% Copy & paste your code from LAB1 here









hold on % This line makes the figure to show all trajectory of the manipulator
end

```

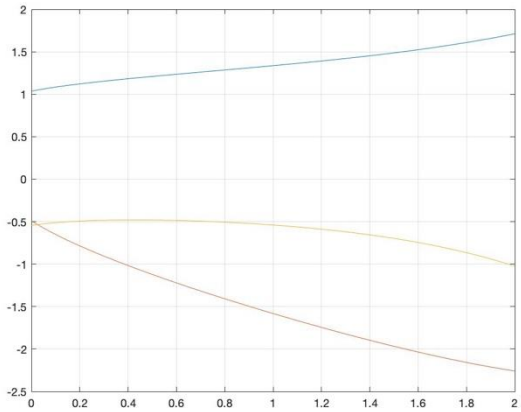
Copy & paste a portion of your LAB01_planar3Rrobot_inversekinematics.m into plotManipulator.m so that this function plots the 3R manipulator configuration given information (q, L_1, L_2, L_3).

Part D: End Effector Trajectory Planning - Arc

Now you will make the 3R manipulator's end effector follow a 90 degree arc with the center of the circle $(x_R, y_R) = (0.1, 0.25)$ and the radius $R = 0.25$.

- Open 'LAB02_ArcTrajectory.m'
- Fill out the blank portion of the code to complete the code.
- If your code runs successfully, following two figures below will be appeared. Left one is manipulator configuration trajectory and the right one is joint angles changes during the simulation time(0-2s)

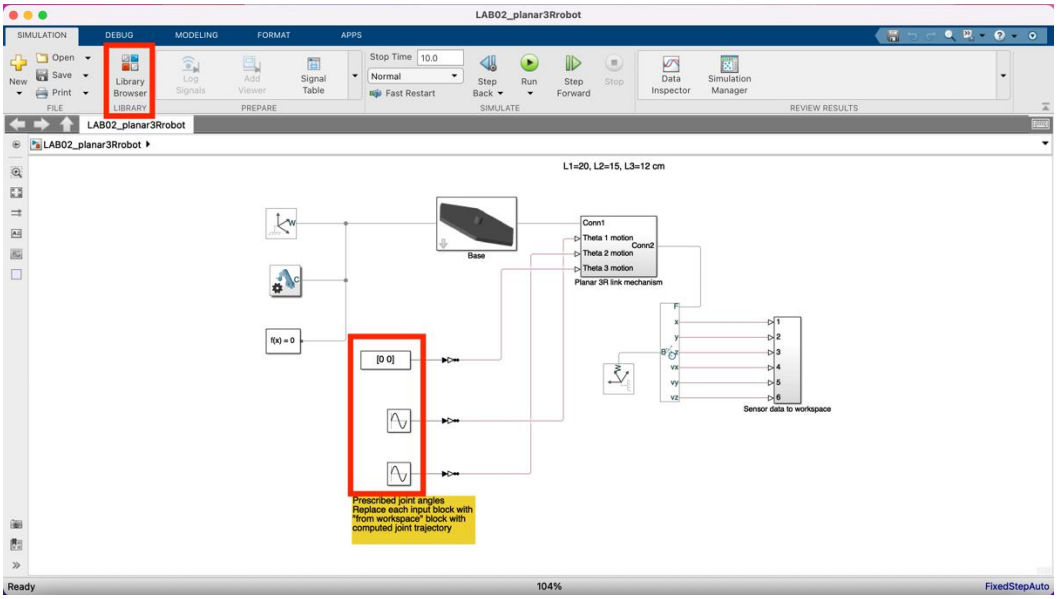




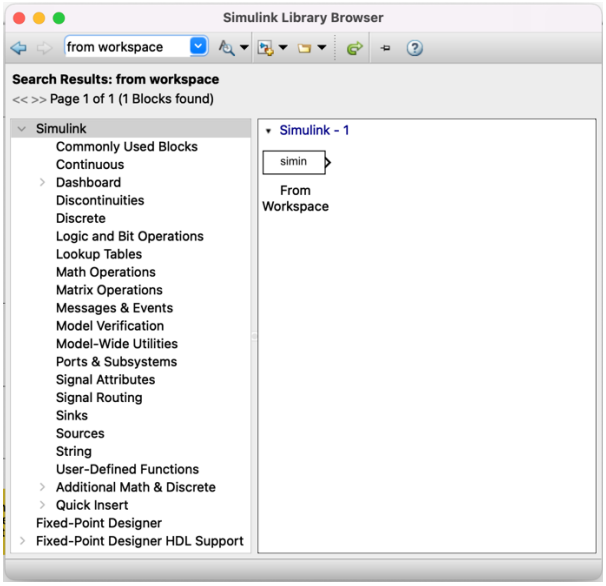
Part E: Simscape Model Modification and Simulation

-Open ‘Lab02_planar3Rrobot.slx’ file. This file is the same one as the ‘Lab01_plana3Rrobot.slx’, just a different name.

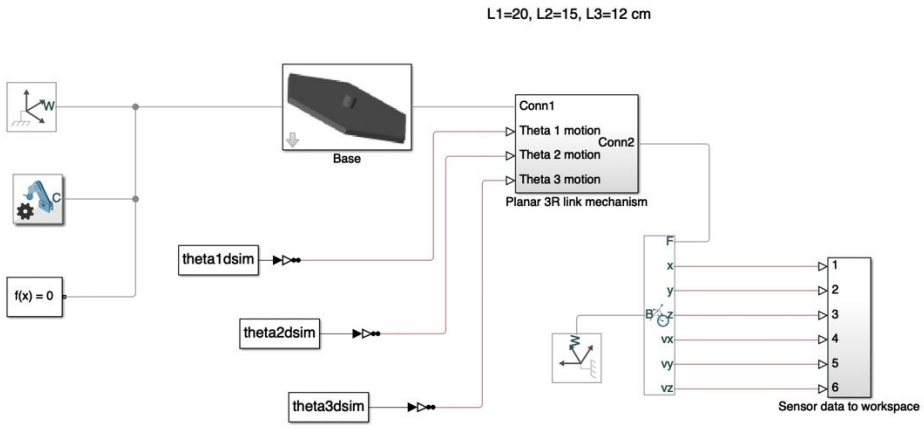
- Now we will replace inputs to three joint angles with ‘from workspace’ block by searching it from the Library Browser.



- Search ‘from workspace’ and click and drag the block to ‘LAB02_planar3Rrobot’. Connect each to joint angles and rename them to ‘theta1dsim’, ‘theta2dsim’ and ‘theta3dsim’.

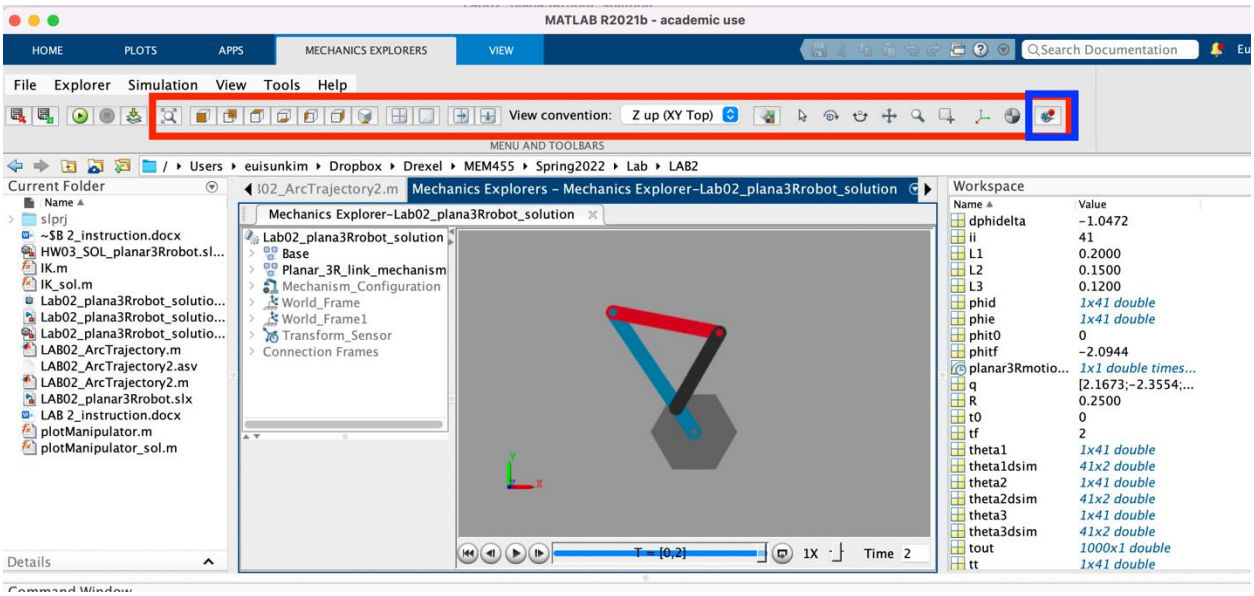


- The result would be look like this below. Now this Simscape model refers to joint angles from the ‘theta1dsim’, ‘theta2dsim’ and ‘theta3dsim’ in the workspace for simulation.



- Run ‘LAB02_ArcTrajectory’ first and make sure ‘theta1dsim’, ‘theta2dsim’ and ‘theta3dsim’ are in the workspace. Then run ‘LAB02_planar3Rrobot.slx’

- Use the tools highlighted in red box to adjust your simulation model so that you can clearly observe the motion of the end effector and also the whole manipulator.
- Check if the end effector follows the arc as we planned. If it follows the arc, click the ‘record’ button on the far right and save the simulation video.



Part F. Create another trajectory that makes the end effector follow straight line that you determine.

- Repeat Part C & Part D with this new trajectory