University of Nevada Las Vegas
Department of Mechanical Engineering
ME 320-1001
Dynamics of Machine

Project 1 Description: Analysis of a 4-Bar Mechanism and Planar 3R Manipulator

Prepared for: Jameson Lee

Prepared by:
Allam, Numan
Bostic, Teague D.
Escobedo, David
Paulino, Reu

Due Date: 4/23/2018

1 Abstract

In this report, an investigation is carried out by the 4-member team to provide a concise analysis on the 4-bar mechanism and the Planer 3R manipulator. The investigation involves the understanding and writing of a MATLAB code based on the “skeleton” code given. The main objective includes completing the “skeleton” code to develop an adaptable code that will solve and plot position of a 4-bar mechanism and 3R manipulator. Also, solve and plot velocity and acceleration of a 4-bar mechanism.

The project consisted of two parts. The first part involves the full analysis of the 4-bar mechanism which includes plotting position, velocity and acceleration. The second part involves the analysis of the planer 3R manipulator, which includes plotting position. Each parts’ problems were analytically resolved by the completed “skeleton” code. The equations for position, velocity, and acceleration were derived using the vector loop method.

The written code in this project was broken into 4-parts: Grashof Sanity, Mechanism, Path Plan and Links. Grashof Sanity class determines the type of 4-bar mechanism based on lengths passed. The Link class updates the start and end points of the links, stores that data and creates an overlay around the links. The Mechanism class solves for the unknowns of each mechanism type. The Path Plan class constructs the mechanisms’ links depending on the type of mechanism.

2 Introduction:

The planer 4-bar crank-slider mechanism consists of four-interconnected links known as "*crank*", "*coupler*", "*slider*", and the "*ground.*" Once the *crank-slider* is put into motion, the *crank* undergoes the pure rotation while the *slider* (also known as *follower*) undergoes the pure translation and the *coupler* engaged in complex motion. The *ground* component of this mechanism remains strictly static. The mathematical equation for this mechanism is a loop equation which can be defined using the Euler's equation. The planar mechanisms are restricted to the motion defined in two-dimensional space; the project is based on planar mechanism as seen in *Figure 2.1* where the path of rotation is seen as projected by the dotted lines.

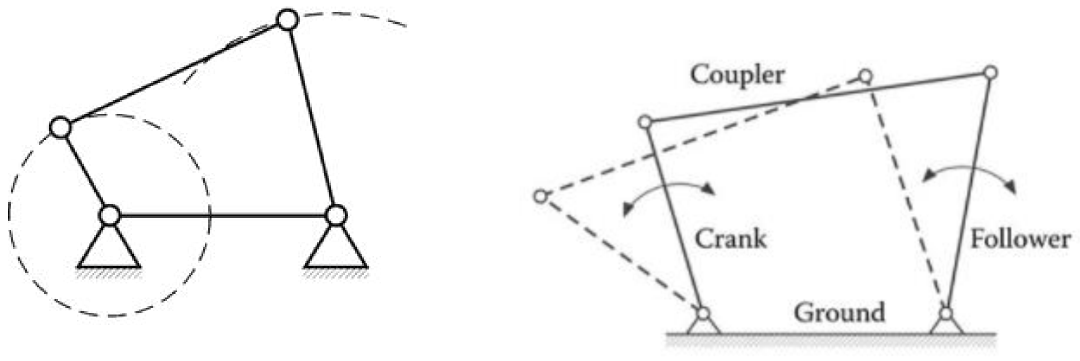


Figure 2.1 Links rotation of Grashof mechanism. This a crank-rocker mechanism.

In a four-bar mechanism, the *Grashof criteria* is used to determine the rotational behavior of the link, in which, these *Grashof standards* are based on the lengths of the crank, coupler, follower, and the ground. In *Table 1* the classifications of the Grashof and non-Grashof mechanisms are included with equation of link-length relationships.

Grashof and Non-Grashof mechanism		
Grashof Type	***Link-Length Relationship***	***Shortest Link***
Crank-rocker	$S + L < P + Q$	Crank
Double-crank	$S + L < P + Q$	Ground
Double rocker	$S + L < P + Q$	Coupler
Change Point	$S + L = P + Q$	Coupler does not rotate
Non-Grashof mechanism		
Triple-rocker	$S + L > P + Q$	Any

Table 2.1 Links rotation of the Grashof and non-Grashof mechanism

In this project, the MATLAB code identified that the mechanism of the four-bar planar mechanism is indeed a “crank-rocker” mechanism, where the shortest link is the crank and only the crank link undergoes a complete rotation. The Grashof type for this mechanism can easily be computed with equation **(2.1)**:

$$S + L < P + Q, \tag{2.1}$$

The vector loop position equation of this mechanism is simply derived as:

$$r_1 e^{i\theta_1} + r_2 e^{i\theta_2} = r_3 e^{i\theta_3} + r_4 e^{i\theta_4} \;, \tag{2.2}$$

The following Euler equation represents the motion of the mechanism that is shown in *Figure 2.1*. The physical representation of the vectors is shown in *Figure 2.2*.

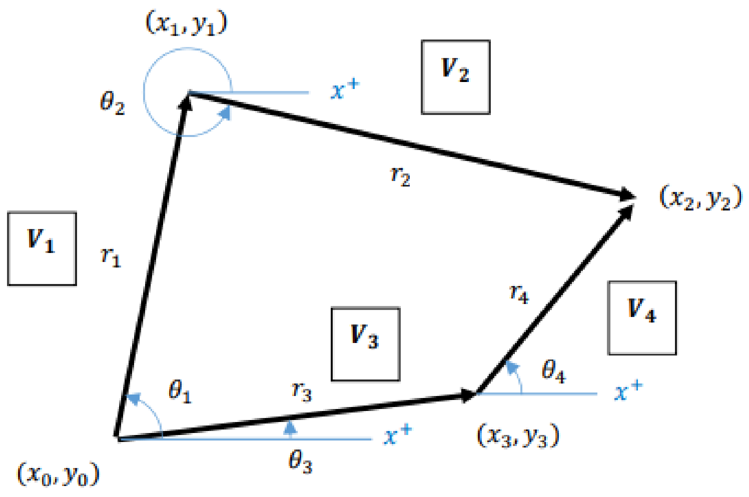


Figure 2.2 Vector representation of the physical mechanism

2.1 Derivation of Position:

Derivation of position of the crank-rocker, crank-crank, and rocker-rocker four-bar mechanisms features in this section. The position equations for the crank-rocker, crank-crank, and rocker-rocker are derived analytically, and all determined position equations will then use to complete the computer code on the MATLAB platform that involves in creating the plots for the position of these mechanism.

The *Matrix Laboratory*, also known as a MATLAB, is a high-level computer language program and a numerical computing environment which will allow to plot the physical response of the position equations of crank-rocker, crank-crank, and rocker-rocker mechanisms. Therefore, it is important to determine the correct position-loop position equation analytically. In succeeding sections of this reports, the crank-rocker, crank-crank, and rocker-rocker positions equations are included.

2.1.1 Four Bar Mechanism:

Crank-Rocker and Crank-Crank Position: In this section, the analytical solution for the crank-rocker and crank-crank mechanism is seeking from determining the position loop equation, using the sum of four vectors as shown in *Figure 2.2*.

$$r_1e^{i\theta_1} + r_2e^{i\theta_2} = r_3e^{i\theta_3} + r_4e^{i\theta_4} \tag{2.3}$$

$$r_1c_1 + r_2c_2 = r_3c_3 + r_4c_4$$

$$r_1s_1 + r_2s_2 = r_3s_3 + r_4s_4$$

$$(r_2c_2)^2 = (r_3c_3 + r_4c_4 - r_1c_1)^2$$

$$(r_2s_2)^2 = (r_3s_3 + r_4s_4 - r_1s_1)^2$$

$$(r_2c_2)^2 + (r_2s_2)^2 = (r_3c_3 + r_4c_4 - r_1c_1)^2 + (r_3s_3 + r_4s_4 - r_1s_1)^2$$

$$r_2{}^2 = r_3{}^2 + r_4{}^2 + r_1{}^2 + 2r_3r_4c_3c_4 - 2r_1r_4c_1c_4 - 2r_3r_1c_3c_1 + 2r_3r_4s_3s_4 - 2r_1r_4s_1s_4 - 2r_3r_1s_3s_1$$

$$r_2{}^2 - r_3{}^2 - r_4{}^2 - r_1{}^2 + 2r_3r_1(c_3c_1 + s_3s_1) = 2r_4(r_3c_3 - r_1c_1)c_4 + 2r_4(r_3s_3 - r_1s_1)\, s_4$$

$$r_2{}^2 - r_3{}^2 - r_4{}^2 - r_1{}^2 + 2r_3r_1\cos(\theta_3 - \theta_1) = 2r_4(r_3c_3 - r_1c_1)c_4 + 2r_4(r_3s_3 - r_1s_1)\, s_4$$

$$C_4 = r_2{}^2 - r_3{}^2 - r_4{}^2 - r_1{}^2 + 2r_3r_1\cos(\theta_3 - \theta_1)$$

$$A_4 = 2r_4(r_3c_3 - r_1c_1)$$

$$B_4 = 2r_4(r_3s_3 - r_1s_1)$$

$$C_4 = A_4 c_4 + B_4 s_4$$

$$C_4 = A_4\left(\frac{1-\tan^2\frac{\theta_4}{2}}{1+\tan^2\frac{\theta_4}{2}}\right) + B_4\left(\frac{2\tan\frac{\theta_4}{2}}{1+\tan^2\frac{\theta_4}{2}}\right)$$

$$C_4\left(1+\tan^2\frac{\theta_4}{2}\right) = A_4\left(1-\tan^2\frac{\theta_4}{2}\right) + B_4\left(2\tan\frac{\theta_4}{2}\right)$$

$$(A_4 + C_4)\tan^2\frac{\theta_4}{2} - 2B_4\tan\frac{\theta_4}{2} + (C_4 - A_4) = 0$$

$$\tan\frac{\theta_4}{2} = \frac{2B_4 \pm \sqrt{4{B_4}^2 - 4(A_4+C_4)(C_4-A_4)}}{2(A_4+C_4)} = \frac{B_4 \pm \sqrt{{B_4}^2 + {A_4}^2 - {C_4}^2}}{(A_4+C_4)}$$

$$\theta_4 = 2\tan^{-1}\left(\frac{B_4 \pm \sqrt{{B_4}^2 + {A_4}^2 - {C_4}^2}}{(A_4+C_4)}\right) \tag{2.4}$$

Repeating the similar steps for computing the θ_2. It is essential to Isolate the θ_4.

$$r_1c_1 + r_2c_2 - r_3c_3 = r_4c_4$$

$$r_1s_1 + r_2s_2 - r_3s_3 = r_4s_4$$

$$C_2 = {r_4}^2 - {r_3}^2 - {r_2}^2 - {r_1}^2 + 2r_1r_3\cos(\theta_1 - \theta_3)$$

$$A_2 = 2r_2(r_1c_1 - r_3c_3)$$

$$B_2 = 2r_2(r_1s_1 - r_3s_3)$$

$$C_2 = A_2c_2 + B_2s_2$$

$$\theta_2 = 2\tan^{-1}\left(\frac{B_2 \pm \sqrt{{B_2}^2 + {A_2}^2 - {C_2}^2}}{(A_2 + C_2)}\right) \tag{2.5}$$

Rocker-Rocker Position:

In this section, the analytical position equation for the for the rocker-rocker is based on summing the sum of four vectors as shown in *Figure 2.2*.

Repeating the approach introduce in computing the position equation for the Crank-Rocker to determine the value of θ_1. By Isolating the θ_4, the equation corresponds to

$$r_1 e^{i\theta_1} + r_2 e^{i\theta_2} = r_3 e^{i\theta_3} + r_4 e^{i\theta_4} \tag{2.3}$$

$$r_1 c_1 + r_2 c_2 - r_3 c_3 = r_4 c_4$$

$$r_1 s_1 + r_2 s_2 - r_3 s_3 = r_4 s_4$$

$$C_1 = {r_4}^2 - {r_3}^2 - {r_2}^2 - {r_1}^2 + 2 r_2 r_3 \cos(\theta_3 - \theta_2)$$

$$A_1 = 2r_1(r_2 c_2 - r_3 c_3)$$

$$B_1 = 2r_1(r_2 s_2 - r_3 s_3)$$

$$C_1 = A_1 c_2 + B_1 s_2$$

$$\theta_1 = 2\tan^{-1}\left(\frac{B_1 \pm \sqrt{{B_1}^2 + {A_1}^2 - {C_1}^2}}{(A_1 + C_1)}\right) \tag{2.6}$$

Repeating the similar approach to determine the value of the θ_4 by isolating the θ_1 , the corresponding equations are:

$$r_4 c_4 + r_3 c_3 - r_2 c_2 = r_1 c_1$$

$$r_4 s_4 + r_3 s_3 - r_2 s_2 = r_1 s_1$$

$$C_4 = {r_1}^2 - {r_3}^2 - {r_2}^2 - {r_4}^2 + 2 r_2 r_3 \cos(\theta_2 - \theta_3)$$

$$A_4 = 2 r_4 (r_3 c_3 - r_2 c_2)$$

$$B_4 = 2 r_4 (r_3 s_3 - r_2 s_2)$$

$$C_4 = A_4 c_2 + B_4 s_2$$

$$\theta_4 = 2 \tan^{-1}\left(\frac{B_4 \pm \sqrt{{B_4}^2 + {A_4}^2 - {C_4}^2}}{(A_4 + C_4)}\right) \tag{2.7}$$

2.1.2 3R Manipulator:

Crank-Slider Position: In this section, the analytical solution for the crank-rocker is determined by summing the three vectors, which will help towards determining the position loop equation for the Crank-Rocker.

By summing the physical vectors, the position equation for the Crank-Rocker can be determined, that will be used to compute the value of the θ_1. This unknown value for the θ_1 can be solved by isolating the θ_2 in position loop equation as the approach seen below:

$$r_1 e^{i\theta_1} + r_2 e^{i\theta_2} = r_3 e^{i\theta_3} \tag{2.8}$$

$$r_2 c_2 = r_3 c_3 - r_1 c_1$$

$$r_2 s_2 = r_3 s_3 - r_1 s_1$$

$$C_1 = {r_1}^2 + {r_3}^2 - {r_2}^2$$

$$A_1 = 2r_1(r_3 c_3)$$

$$B_1 = 2r_1(r_3 s_3)$$

$$C_1 = A_1 c_1 + B_1 s_1$$

$$\theta_1 = 2\tan^{-1}\left(\frac{B_1 \pm \sqrt{{B_1}^2 + {A_1}^2 - {C_1}^2}}{(A_1 + C_1)}\right) \tag{2.9}$$

Repeating the similar approach for the Crank-Rocker to find θ_1, it is important to isolate θ_2.

$$r_1 e^{i\theta_1} + r_2 e^{i\theta_2} = r_3 e^{i\theta_3} \tag{2.8}$$

$$r_1 c_1 = r_3 c_3 - r_2 c_2$$

$$r_1 s_1 = r_3 s_3 - r_2 s_2$$

$$C_2 = {r_2}^2 + {r_3}^2 - {r_1}^2$$

$$A_2 = 2r_2(r_3 c_3)$$

$$B_1 = 2r_2(r_3 s_3)$$

$$C_2 = A_2 c_2 + B_2 s_2$$

$$\theta_2 = 2\tan^{-1}\left(\frac{B_2 \pm \sqrt{{B_2}^2 + {A_2}^2 - {C_2}^2}}{(A_2 + C_2)}\right) \qquad (2.10)$$

2.2 Derivation of Velocity:

In this section, the discussions are offered to properly derive the velocity loop equations for the crank-rocker, crank-crank, and rocker-rocker four-bar mechanisms. All these vector loop equations are derived analytically based on the position-loop equations, using the Product Rule. These analytically determined vector-loop equations will be used to complete the existing computer code written on the MATLAB platform that involves in creating the plots for the velocity plots of these mechanism.

2.2.1 Four Bar Mechanism:

<u>Crank-Rocker and Crank-Crank Velocity</u>:

In this section, the analytical solution for the crank-rocker and the crank-crank mechanism is based on the position loop equations by using the product rule to determine the vector-loop equations as followed. The final answers for vector-loop equations are based on introducing the matrices.

$$i\dot{\theta}_1 r_1 e^{-i\theta_1} + i\dot{\theta}_2 r_2 e^{-i\theta_2} = i\dot{\theta}_4 r_4 e^{-i\theta_4} \qquad (2.7)$$

Note that θ_3 is a constant and therefore $\dot{\theta}_3 = 0$.

$$r_1 c_1 \dot{\theta}_1 = r_4 c_4 \dot{\theta}_4 - r_2 c_2 \dot{\theta}_2$$

$$r_1 s_1 \dot{\theta}_1 = r_4 s_4 \dot{\theta}_4 - r_2 s_2 \dot{\theta}_2$$

$$\begin{bmatrix} r_1c_1\dot{\theta}_1 \\ r_1s_1\dot{\theta}_1 \end{bmatrix} = \begin{bmatrix} r_4c_4 & -r_2c_2 \\ r_4s_4 & -r_2s_2 \end{bmatrix} \begin{bmatrix} \dot{\theta}_4 \\ \dot{\theta}_2 \end{bmatrix}$$

$$A = \begin{bmatrix} r_1c_1\dot{\theta}_1 \\ r_1s_1\dot{\theta}_1 \end{bmatrix} \; B = \begin{bmatrix} r_4c_4 & -r_2c_2 \\ r_4s_4 & -r_2s_2 \end{bmatrix} C = \begin{bmatrix} \dot{\theta}_4 \\ \dot{\theta}_2 \end{bmatrix}$$

$$C = B^{-1}A$$

$$\begin{bmatrix} \dot{\theta}_4 \\ \dot{\theta}_2 \end{bmatrix} = \begin{bmatrix} r_4c_4 & -r_2c_2 \\ r_4s_4 & -r_2s_2 \end{bmatrix}^{-1} \begin{bmatrix} r_1c_1\dot{\theta}_1 \\ r_1s_1\dot{\theta}_1 \end{bmatrix} \tag{2.8}$$

<u>Rocker-Rocker Velocity:</u>

In this section, the analytical vector loop equation for the rocker-rocker corresponds to:

$$i\dot{\theta}_1 r_1 e^{-i\theta_1} + i\dot{\theta}_2 r_2 e^{-i\theta_2} = i\dot{\theta}_4 r_4 e^{-i\theta_4}$$

Note that θ_3 is a constant and therefore $\dot{\theta}_3 = 0$.

$$r_2c_2\dot{\theta}_2 = r_4c_4\dot{\theta}_4 - r_1c_1\dot{\theta}_2$$

$$r_2s_2\dot{\theta}_2 = r_4s_4\dot{\theta}_4 - r_1s_1\dot{\theta}_1$$

$$\begin{bmatrix} r_2c_2\dot{\theta}_2 \\ r_2s_2\dot{\theta}_2 \end{bmatrix} = \begin{bmatrix} r_4c_4 & -r_1c_1 \\ r_4s_4 & -r_1s_1 \end{bmatrix} \begin{bmatrix} \dot{\theta}_4 \\ \dot{\theta}_1 \end{bmatrix}$$

$$A = \begin{bmatrix} r_2c_2\dot{\theta}_2 \\ r_2s_2\dot{\theta}_2 \end{bmatrix} \; B = \begin{bmatrix} r_4c_4 & -r_1c_1 \\ r_4s_4 & -r_1s_1 \end{bmatrix} C = \begin{bmatrix} \dot{\theta}_4 \\ \dot{\theta}_1 \end{bmatrix}$$

$$C = B^{-1}A$$

$$\begin{bmatrix}\dot{\theta}_4\\ \dot{\theta}_1\end{bmatrix} = \begin{bmatrix} r_4c_4 & -r_1c_1\\ r_4s_4 & -r_1s_1\end{bmatrix}^{-1}\begin{bmatrix} r_2c_2\dot{\theta}_2\\ r_2s_2\dot{\theta}_2\end{bmatrix} \tag{2.9}$$

2.3 Derivation of Acceleration:

This section offers discussion on determining the acceleration loop equations for the crank-rocker, crank-crank, and rocker-rocker four-bar mechanisms. For all those mechanisms, the acceleration equations are derived analytically based on the vector-loop equations, using the Product Rule and the Chain Rule. These analytically determined acceleration-loop equations will be used to complete the existing computer code written on the MATLAB platform that involves in creating the plots for the acceleration plots of these mechanism.

2.3.1 Four Bar Mechanism

Crank-Rocker and Crank-Crank acceleration: In this section, the analytical solution for the crank-rocker and the crank-crank mechanism is based on the vector-loop equations by using the chain rule, with combination of the product rule when needed, to determine the acceleration-loop equations as followed. The final answers for acceleration loop equations are also based on introducing the matrices.

$$\left(i\ddot{\theta}_1 - \dot{\theta}_1^{\,2}\right) r_1 e^{-i\theta_1} + \left(i\ddot{\theta}_2 - \dot{\theta}_2^{\,2}\right) r_2 e^{-i\theta_2} = \left(i\ddot{\theta}_4 - \dot{\theta}_4^{\,2}\right) r_4 e^{-i\theta_4} \tag{2.20}$$

$$r_1\left(-\ddot{\theta}_1 s_1 - \dot{\theta}_1^{\,2} c_1\right) = r_4\left(-\ddot{\theta}_4 s_4 - \dot{\theta}_4^{\,2} c_4\right) - r_2\left(-\ddot{\theta}_2 s_2 - \dot{\theta}_2^{\,2} c_2\right)$$

$$r_1\left(\ddot{\theta}_1 c_1 - \dot{\theta}_1^{\,2} s_1\right) = r_4\left(\ddot{\theta}_4 c_4 - \dot{\theta}_4^{\,2} s_4\right) - r_2\left(\ddot{\theta}_2 c_2 - \dot{\theta}_2^{\,2} s_2\right)$$

$$r_1\ddot{\theta}_1 s_1 + r_1\dot{\theta}_1{}^2 c_1 - r_4\dot{\theta}_4{}^2 c_4 + r_2\dot{\theta}_2{}^2 c_2 = r_4\ddot{\theta}_4 s_4 - r_2\ddot{\theta}_2 s_2$$

$$r_1\ddot{\theta}_1 c_1 - r_1\dot{\theta}_1{}^2 s_1 + r_4\dot{\theta}_4{}^2 s_4 - r_2\dot{\theta}_2{}^2 s_2 = r_4\ddot{\theta}_4 c_4 - r_2\ddot{\theta}_2 c_2$$

$$\begin{bmatrix} r_1\ddot{\theta}_1 s_1 + r_1\dot{\theta}_1{}^2 c_1 - r_4\dot{\theta}_4{}^2 c_4 + r_2\dot{\theta}_2{}^2 c_2 \\ r_1\ddot{\theta}_1 c_1 - r_1\dot{\theta}_1{}^2 s_1 + r_4\dot{\theta}_4{}^2 s_4 - r_2\dot{\theta}_2{}^2 s_2 \end{bmatrix} = \begin{bmatrix} r_4 s_4 & -r_2 s_2 \\ r_4 c_4 & -r_2 c_2 \end{bmatrix} \begin{bmatrix} \dot{\theta}_4 \\ \dot{\theta}_2 \end{bmatrix}$$

$$A = \begin{bmatrix} r_1\ddot{\theta}_1 s_1 + r_1\dot{\theta}_1{}^2 c_1 - r_4\dot{\theta}_4{}^2 c_4 + r_2\dot{\theta}_2{}^2 c_2 \\ r_1\ddot{\theta}_1 c_1 - r_1\dot{\theta}_1{}^2 s_1 + r_4\dot{\theta}_4{}^2 s_4 - r_2\dot{\theta}_2{}^2 s_2 \end{bmatrix} \; B = \begin{bmatrix} r_4 s_4 & -r_2 s_2 \\ r_4 c_4 & -r_2 c_2 \end{bmatrix} \; C = \begin{bmatrix} \dot{\theta}_4 \\ \dot{\theta}_2 \end{bmatrix}$$

$$C = B^{-1} A$$

$$\begin{bmatrix} \dot{\theta}_4 \\ \dot{\theta}_2 \end{bmatrix} = \begin{bmatrix} r_4 s_4 & -r_2 s_2 \\ r_4 c_4 & -r_2 c_2 \end{bmatrix}^{-1} \begin{bmatrix} r_1\ddot{\theta}_1 s_1 + r_1\dot{\theta}_1{}^2 c_1 - r_4\dot{\theta}_4{}^2 c_4 + r_2\dot{\theta}_2{}^2 c_2 \\ r_1\ddot{\theta}_1 c_1 - r_1\dot{\theta}_1{}^2 s_1 + r_4\dot{\theta}_4{}^2 s_4 - r_2\dot{\theta}_2{}^2 s_2 \end{bmatrix} \tag{2.13}$$

Rocker-Rocker Acceleration: In this section, the analytical acceleration loop equation for the rocker-rocker corresponds to:

$$\left(i\ddot{\theta}_1 - \dot{\theta}_1{}^2\right) r_1 e^{-i\theta_1} + \left(i\ddot{\theta}_2 - \dot{\theta}_2{}^2\right) r_2 e^{-i\theta_2} = \left(i\ddot{\theta}_4 - \dot{\theta}_4{}^2\right) r_4 e^{-i\theta_4}$$

$$r_2\left(-\ddot{\theta}_2 s_2 - \dot{\theta}_2{}^2 c_2\right) = r_4\left(-\ddot{\theta}_4 s_4 - \dot{\theta}_4{}^2 c_4\right) - r_1\left(-\ddot{\theta}_1 s_1 - \dot{\theta}_1{}^2 c_1\right)$$

$$r_2\left(\ddot{\theta}_2 c_2 - \dot{\theta}_2{}^2 s_2\right) = r_4\left(\ddot{\theta}_4 c_4 - \dot{\theta}_4{}^2 s_4\right) - r_1\left(\ddot{\theta}_1 c_1 - \dot{\theta}_1{}^2 s_1\right)$$

$$r_2\ddot{\theta}_2 s_2 + r_2\dot{\theta}_2{}^2 c_2 - r_4\dot{\theta}_4{}^2 c_4 + r_1\dot{\theta}_1{}^2 c_1 = r_4\ddot{\theta}_4 s_4 - r_1\ddot{\theta}_1 s_1$$

$$r_2\ddot{\theta}_2 c_2 - r_2\dot{\theta}_2{}^2 s_2 + r_4\dot{\theta}_4{}^2 s_4 - r_1\dot{\theta}_1{}^2 s_1 = r_4\ddot{\theta}_4 c_4 - r_1\ddot{\theta}_1 c_1$$

$$\begin{bmatrix} r_2\ddot{\theta}_2 s_2 + r_2\dot{\theta}_2{}^2 c_2 - r_4\dot{\theta}_4{}^2 c_4 + r_1\dot{\theta}_1{}^2 c_1 \\ r_2\ddot{\theta}_2 c_2 - r_2\dot{\theta}_2{}^2 s_2 + r_4\dot{\theta}_4{}^2 s_4 - r_1\dot{\theta}_1{}^2 s_1 \end{bmatrix} = \begin{bmatrix} r_4 s_4 & -r_1 s_1 \\ r_4 c_4 & -r_1 c_1 \end{bmatrix} \begin{bmatrix} \dot{\theta}_4 \\ \dot{\theta}_1 \end{bmatrix}$$

$$A = \begin{bmatrix} r_2\ddot{\theta}_2 s_2 + r_2\dot{\theta}_2{}^2 c_2 - r_4\dot{\theta}_4{}^2 c_4 + r_1\dot{\theta}_1{}^2 c_1 \\ r_2\ddot{\theta}_2 c_2 - r_2\dot{\theta}_2{}^2 s_2 + r_4\dot{\theta}_4{}^2 s_4 - r_1\dot{\theta}_1{}^2 s_1 \end{bmatrix} B = \begin{bmatrix} r_4 s_4 & -r_1 s_1 \\ r_4 c_4 & -r_1 c_1 \end{bmatrix} C = \begin{bmatrix} \dot{\theta}_4 \\ \dot{\theta}_1 \end{bmatrix}$$

$$C = B^{-1}A$$

$$\begin{bmatrix} \dot{\theta}_4 \\ \dot{\theta}_1 \end{bmatrix} = \begin{bmatrix} r_4 s_4 & -r_1 s_1 \\ r_4 c_4 & -r_1 c_1 \end{bmatrix}^{-1} \begin{bmatrix} r_2\ddot{\theta}_2 s_2 + r_2\dot{\theta}_2{}^2 c_2 - r_4\dot{\theta}_4{}^2 c_4 + r_1\dot{\theta}_1{}^2 c_1 \\ r_2\ddot{\theta}_2 c_2 - r_2\dot{\theta}_2{}^2 s_2 + r_4\dot{\theta}_4{}^2 s_4 - r_1\dot{\theta}_1{}^2 s_1 \end{bmatrix} \tag{2.42}$$

3 Results:

The crank-rocker mechanism results were as expected. The mechanism motion is correct since the crank link, the first link, is the driver and makes a full counterclockwise rotation. Additionally, due to the first link being the driver it is set to a constant angular velocity, thus no acceleration, therefore the graphs for that angle should be straight lines. In the team’s data the angle theta one was indeed a straight line in all the graphs which includes angular displacement, velocity, and acceleration. The rocker-rocker motion plot is also correct. The mechanism motion is correct since the coupler, second link, is the driver and makes a full counterclockwise rotation. In this case, due to the coupler being the driver the angle theta two should be constant in all plots. In the data created the angle theta two is indeed constant in all plots. The next motion is the crank-crank mechanism in which also resulted in the correct results. The motion is correct since the driving link is the first link and the follower link are the fourth link. In the plots the angle theta one has a constant displacement and velocity while the acceleration is zero. For the three bar a desired end effector is the driving link and will move in any configuration wanted. In both the four-bar and the three bar all closures were successfully made. The code is also adaptable in the sense that the ground link can be set to any angle and the link overlay can be increased in size or removed completely. The overarching structure of the code is as follows: the main_4bar and the main_3R are the main classes in which call upon the other classes link, mechanism, pathPlan, and GrashofofSanity. Link sets the foundation for mechanism and pathplan. In the end, the link class was used to find the unknowns in the mechanism class. Then the mechanism class uses the code from the link class and then is applied in the pathPlan class. The pathPlan class uses the link class and the mechanism class to create the actual links of the plots. Lastly, the GrashofSanity class is used to find the type of the mechanism by using the Grashof equation. All of these classes are needed in order for the main_4bar and the main_3R to work. Since, both the main_4bar and the main_3R both work then it can be said that the results were successful.

3.1 Four Bar Mechanism

The main four-bar file is the base file where the parameters are set and calls the supporting files such as GrashofSanity, Link, mechanisms, and pathPlan. The first portion of the main four bar file sets the parameters of the four-bar mechanism. The user will define the lengths of all four links and define the ground angle (th3). Then the user will define the geometry parameters of the link overlay. The user will have the option to turn off the overlay later in the code. The next section of the main four-bar file defines the simulation settings such as simulation time step, number of time steps, a vector for steps through a circle, and a time vector. Also, during this section the closure of the mechanism is defined with closure set to 1 being the first closure and closure set to 2 begin the second closure of the mechanism. The next section, Object Construction, will generate the links of the four-bar mechanism by calling the supporting files. Firstly, the property of all four links are sent to the supporting file *Link*. Link.m is a class file that accepts length, angle, initial x coordinate, and initial y coordinate. Then stores the instantaneous position of the initial and final coordinates of the vector, the orientation of the vector, velocity and acceleration data, and overlay data. Continuing with Object Construction section of the main four-bar file, now that the links have been defined the overlay can be set to true meaning on or false meaning off. If the overlay is set to true, then the overlay shape defined early in the main four-bar file is scaled to the length of each link individually. Next in the Object Construction is to determine the type of four-bar mechanism by passing data to the supporting file *GrashofSanity*. GrashofSanity.m is a class file that accepts a vector of lengths which was defined in the first section of main four-bar file, setting parameters. The GrashofSanity.m is a class file will define three variables typeCode, type, and warning. TypeCode is a numerical valued assigned to a type of four-bar mechanism. TypeCode 0 is type changepoint, TypeCode 1 is type non-Grashof, TypeCode 2 is type crank-rocker, TypeCode 3 is type rocker-rocker, and TypeCode 4 is type crank-crank. If the lengths of the four-bar mechanism are found to be either typecode 0 or 1, a warning is assigned and displayed in the command window, and in the main four-bar file an error is executed stopping the code from going any further. If a typeCode of 2 or 4 is assigned, then in the main four-bar file the driving link is set to link 1 and the angular velocity of the driving link is set. If a typeCode of 3 is assigned, then in the main four-bar file the driving link is set to link 2 and the angular velocity of the driving link is set. Next the code will send the type to support file *mechanism* to define initiate the correct if statement. Mechanism.m class accepts type, closure, known lengths, and known angles. Then solves for the unknowns of the mechanism. Since crank-rocker and crank-crank have different unknowns than a rocker-rocker mechanism the Mechanism.m class compares the Type determined in GrashofSanity.m class to if statements to solve for the correct unknowns. Both closures are solved for instantaneously in the Mechanism.m class. Lastly in the object construction section of main four-bar file the code will create a structure P of type *pathPlan* and pass the support file *pathPlan* all the links created, mechanism, closure, and any points of interest which for a four bar we just them to (0,0) as it is not used. pathPlan.m class accepts links, mechanism, mechtype, closure, and desired position of interest points. Then defines the closure in mechanism to the closure passed to it and initiates the c.update function. The c.update function in *pathPlan* compares the mechtype to a set of if statements to determine whether the mechanism is a four-bar or a 3R manipulator. The function c.update will then send mechanism the link lengths and angles, and solve for the unknown variables. Next, depending the mechanism type the function c.update will construct the mechanism by defining the initial points and angle of each link vector. For the four-bar mechanism the function c.update will also pass mechanism.m class the known angular velocity and acceleration values and then have mechanism.m class solve for the unknown values for angular velocity and acceleration. If the type does not match any of the if statements the following message will print in the command window "The MECHANISM assigned has not been built in this solver," where MECHANISM is the type of mechanism defined by GrashofSanity.m class. Next section of main four-bar file, Solution section, is the main loop that plots and solves for all positions as the driving link is rotated at the defined velocity. The solution section is a for loop that runs till number of time steps defined early is achieved. The for loop will update the driving link's angle of structure P for each time step. Next the for loop will store and plot the data of all the links. Finally, the for loop will plot the free join which is the free travel points not attached to the ground link. The last section of the main four-bar file plots the angular displacement, velocity, and acceleration of the four-bar mechanism.

3.1.1 Decision Tree for Grashof Mechanism

In the Grashof code file the typecode variable is what is needed along with the text of what type of mechanism it is. The Grashof decision tree starts as follows. If all links are the same or there are two longest and two shortest links then the mechanism is a change point, therefore typecode is equal to zero. If this is not the case then if there are more than one shortest or longest lengths then the mechanism is non-grashof, therefore a typecode of one is given. After this it is tested for Grashof if the Grashof equation is true meaning if the summation of the smallest and largest links minus the summation of the other two links is less than zero, then a switch case is presented to determine which link is smallest. If the smallest link is the crank then the type is a crank-rocker, typecode equals 2. If the smallest length is the coupler then the type is a rocker-rocker typecode equals 3. If the ground is the shortest length, then the type is a crank-crank typecode equals 4. Otherwise, if the summation of the smallest and largest links is equal to the summation of the other two links then the mechanism is change point, thus typecode is again equal to zero. Finally, if that is not the case then the type is non-grashof, typecode equals one.

For example, for the following lengths:

Len = [1 1 1 1]
The code returns:
"This is a Change Point Four Bar Mechanism. All inversions are the same. This type is not assigned."

Len = [.1 .1 .5 .6]
The code returns:
"This is a Non-Grashof Four Bar Mechanism. Non-predictable behavior. This type is not assigned."

Len = [.1 .1 .6 .6]
The code returns:

"This is a Change Point Four Bar Mechanism. All inversions are the same. This type is not assigned."

Len = [.2 .3 .4 .5]
The code returns:
"This is a Change Point Four Bar Mechanism. All inversions are the same. This type is not assigned."

Len= [.2 .3 .4 .6]
The code returns:
"This is a Non-Grashof Four Bar Mechanism. Non-predictable behavior. This type is not assigned."

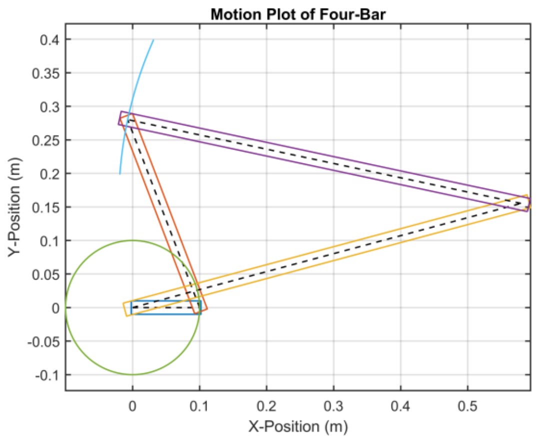
Len = [.1 .3 .6 .6]
The code returns:
"This is a Crank-Rocker Four Bar Mechanism." For Len = [.2 .4 .5 .6] the code returns "This is a Crank-Rocker Four Bar Mechanism."

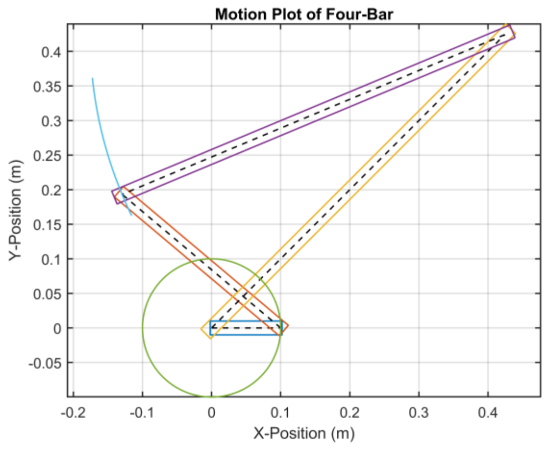
Len = [.3 .1 .4 .5]
The code returns:
"This is a Rocker-Rocker Four Bar Mechanism. Actuator should be on the coupler."

Len = [.5 .2 .05 .4]
The code returns:
"This is a Crank-Crank (Drag-Link) Four Bar Mechanism."

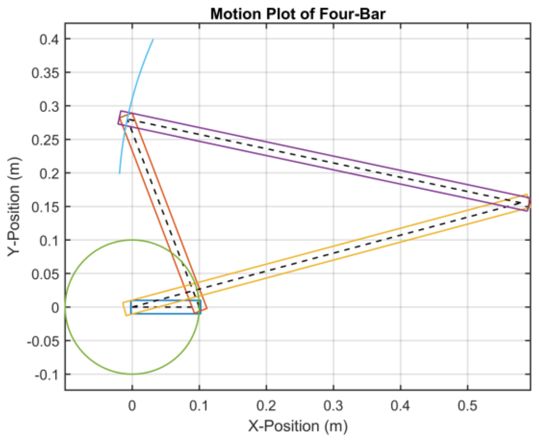
3.1.2 Adaptability in Code

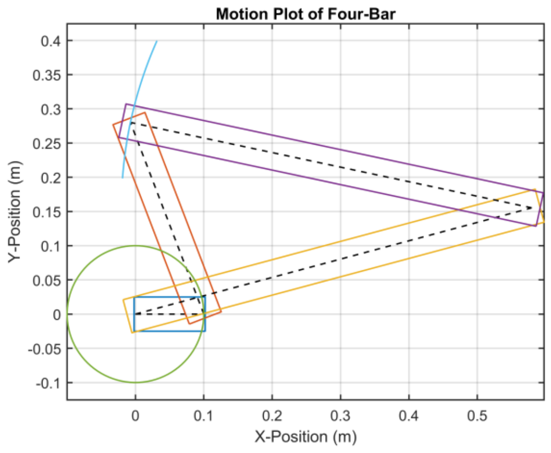
For all plottable configuration the ground link (th3) is valid for any angle. The following plots show the default angle th3 set to pi/12 and the angle th3 changed to pi/2.



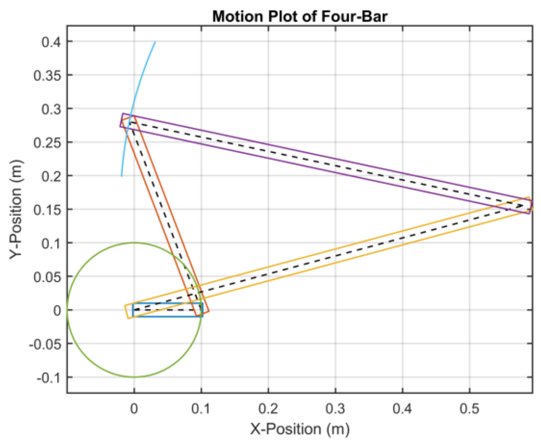
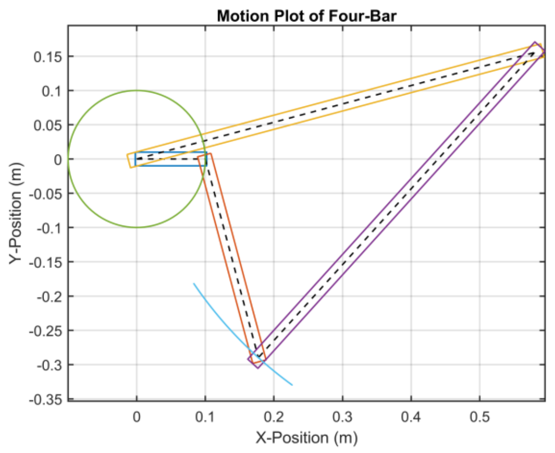
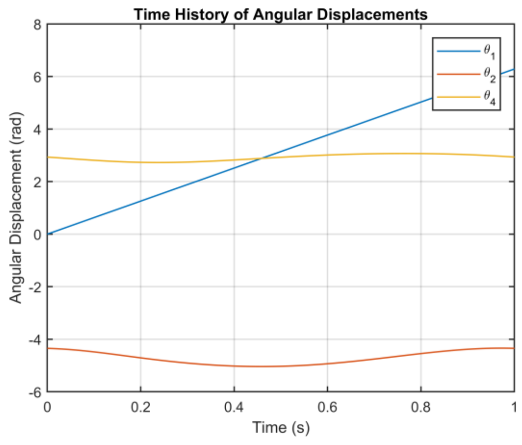
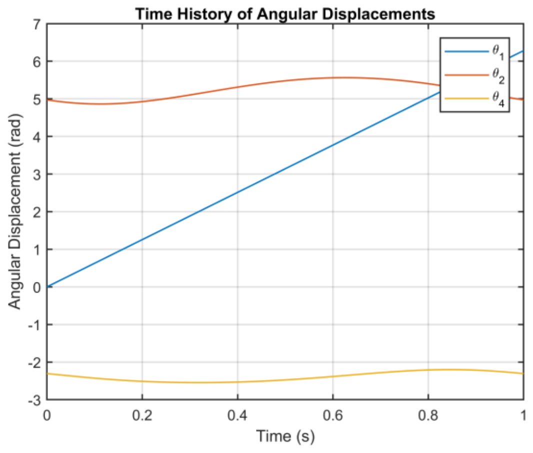
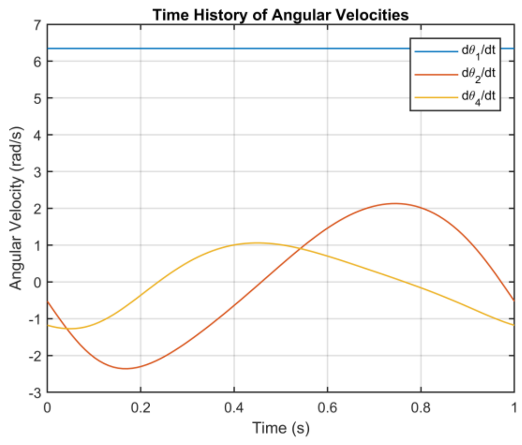
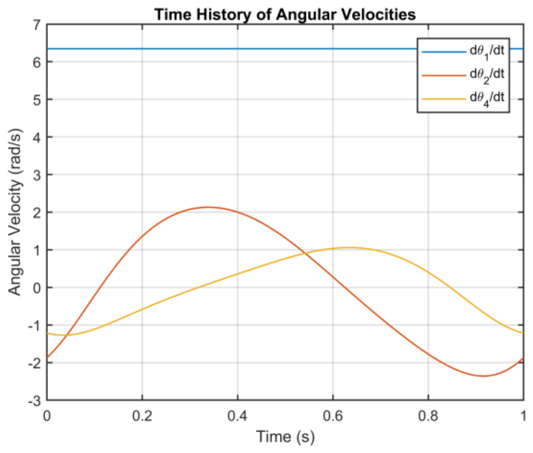


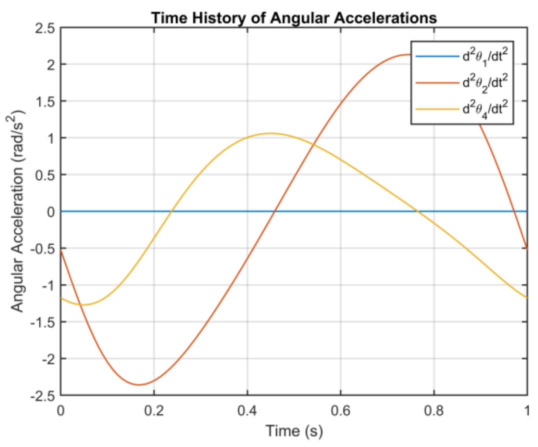
For all plottable configurations the link shape will change if the width of the link (Wid) is changed and/or box_x and box_y local link is changed. The following plots show the default link shape and the width of the link increased.

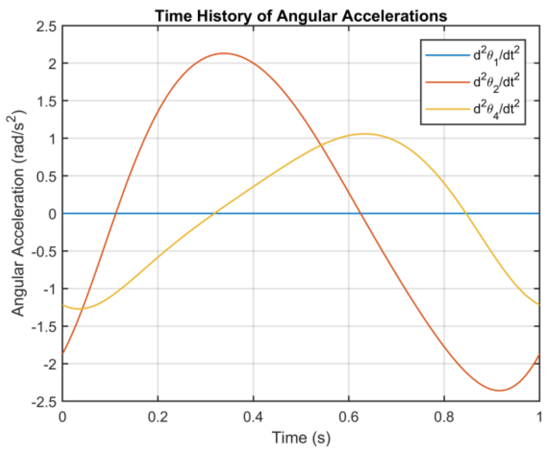




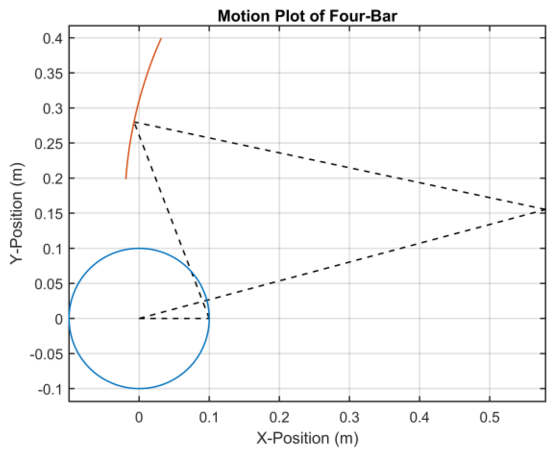
For all plottable configurations both closures are plottable. The following plots show closure 1 displayed on the left and closure two displayed on the right.

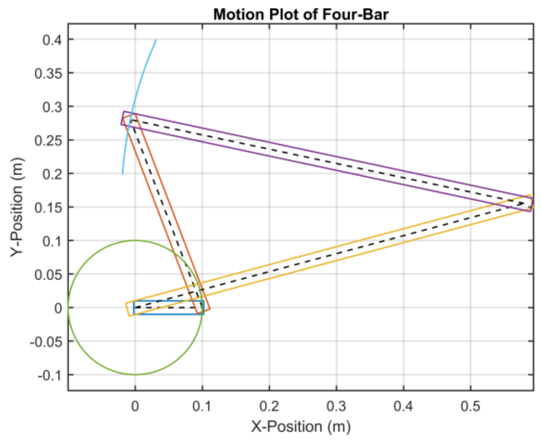

Motion Plot of Four-Bar
X-Position (m)
Y-Position (m)

Motion Plot of Four-Bar
X-Position (m)
Y-Position (m)

Time History of Angular Displacements
θ_1
θ_2
θ_4
Time (s)
Angular Displacement (rad)

Time History of Angular Displacements
θ_1
θ_2
θ_4
Time (s)
Angular Displacement (rad)

Time History of Angular Velocities
$d\theta_1/dt$
$d\theta_2/dt$
$d\theta_4/dt$
Time (s)
Angular Velocity (rad/s)

Time History of Angular Velocities
$d\theta_1/dt$
$d\theta_2/dt$
$d\theta_4/dt$
Time (s)
Angular Velocity (rad/s)





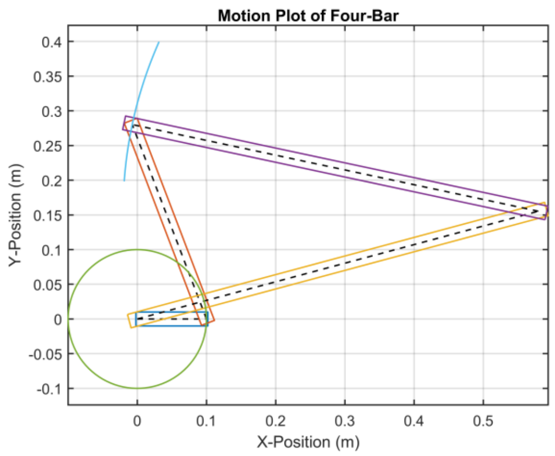
For all plottable configurations the overlay can be toggled on and off by setting overlay equal to true for on, and false for off. The following plots show the overlay set to false on the left and the overlay set to true on the right.

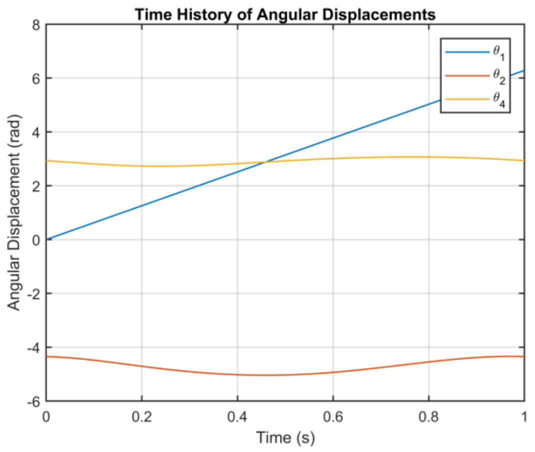


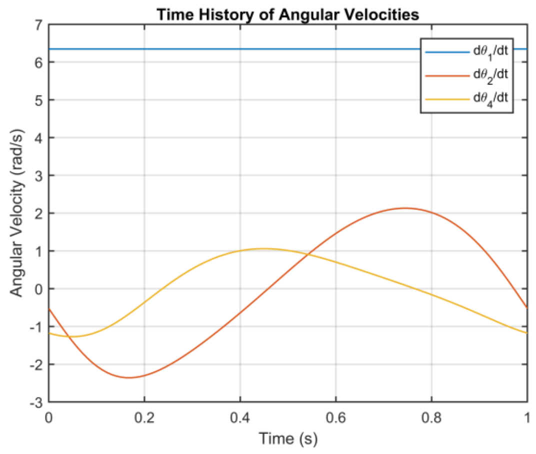


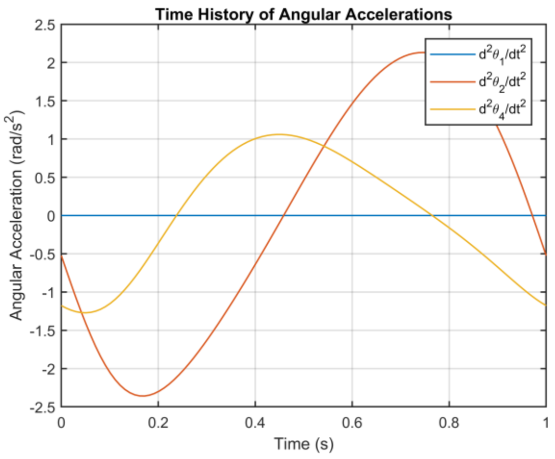
3.1.3 Figures for all Plottable Configurations

The following four graphs show the motion, angular displacement, angular velocity, and angular acceleration plots of a Crank-Rocker Mechanism that has lengths of 0.1, 0.3, 0.6, and 0.6 units, with the 3rd (ground link) and 4th (follower link) being the longest.

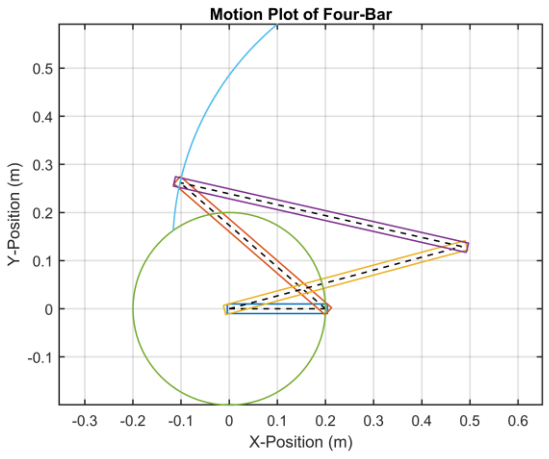


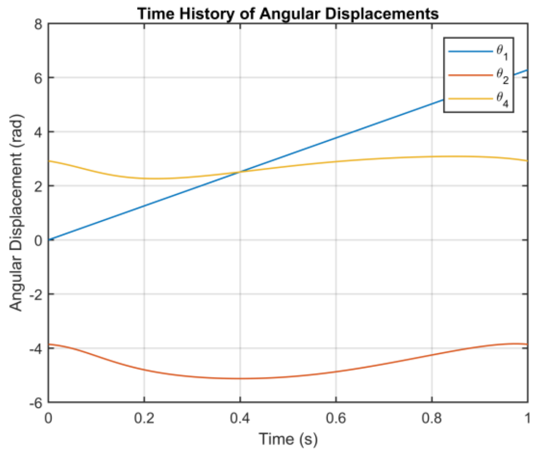


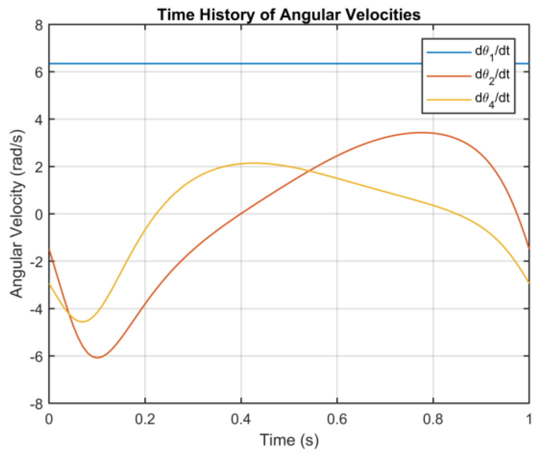


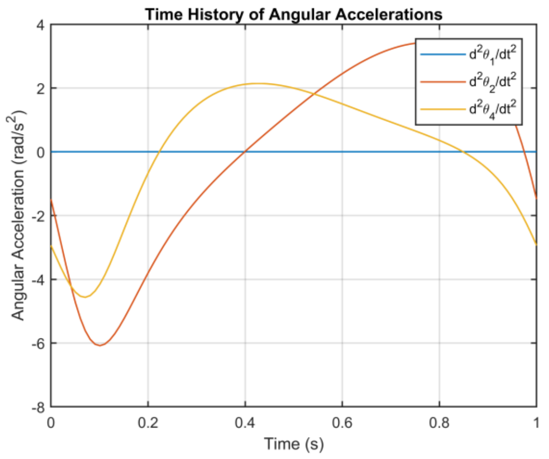


The following four graphs show the motion, angular displacement, angular velocity, and angular acceleration plots of a Crank-Rocker Mechanism that has lengths of 0.2, 0.4, 0.5, and 0.6 units, with the 1st link (drive link) being the shortest.

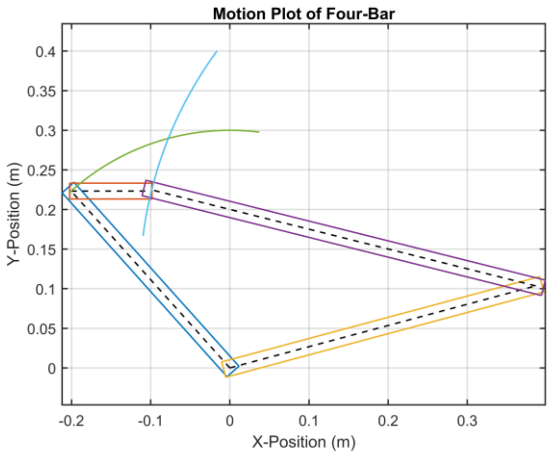

Motion Plot of Four-Bar
Y-Position (m)
X-Position (m)
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.3
-0.2
-0.1
0
0.1
0.2
0.3
0.4
0.5
0.6

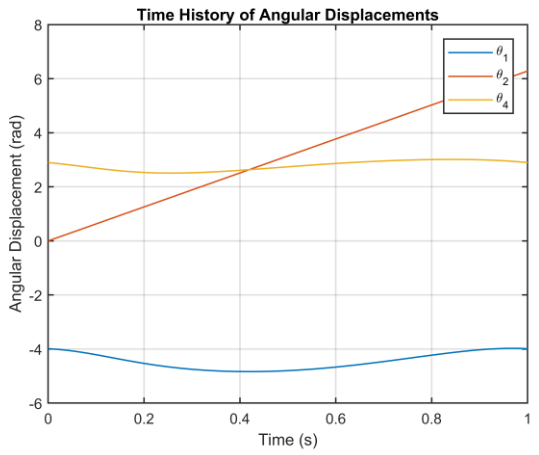

Time History of Angular Displacements
Angular Displacement (rad)
Time (s)
θ_1
θ_2
θ_4
8
6
4
2
0
-2
-4
-6
0
0.2
0.4
0.6
0.8
1

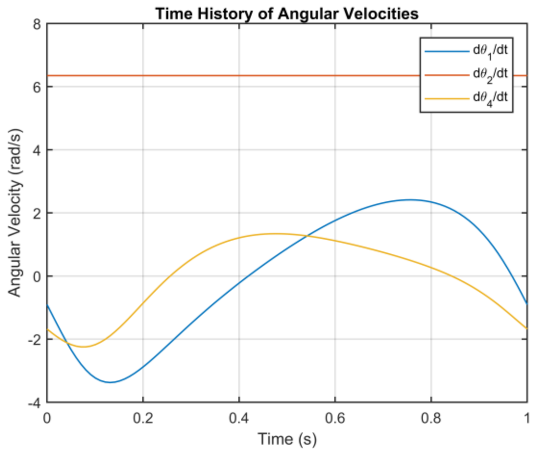

Time History of Angular Velocities
Angular Velocity (rad/s)
Time (s)
$d\theta_1/dt$
$d\theta_2/dt$
$d\theta_4/dt$
8
6
4
2
0
-2
-4
-6
-8
0
0.2
0.4
0.6
0.8
1

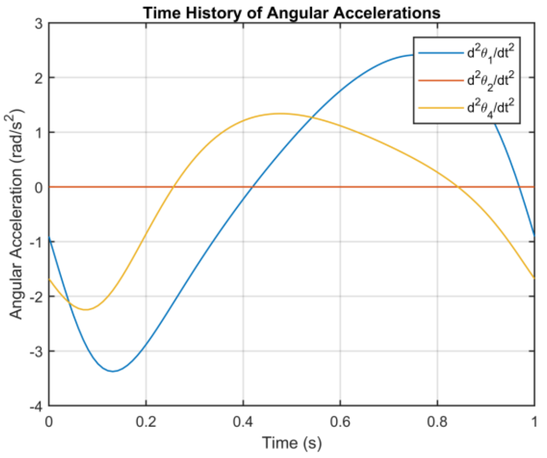

Time History of Angular Accelerations
Angular Acceleration (rad/s^2)
Time (s)
$d^2\theta_1/dt^2$
$d^2\theta_2/dt^2$
$d^2\theta_4/dt^2$
4
2
0
-2
-4
-6
-8
0
0.2
0.4
0.6
0.8
1

The following four graphs show the motion, angular displacement, angular velocity, and angular acceleration plots of a Rocker-Rocker Mechanism that has lengths of 0.3, 0.1, 0.4, and 0.5 units, with the 2nd link (coupler link) being the shortest.

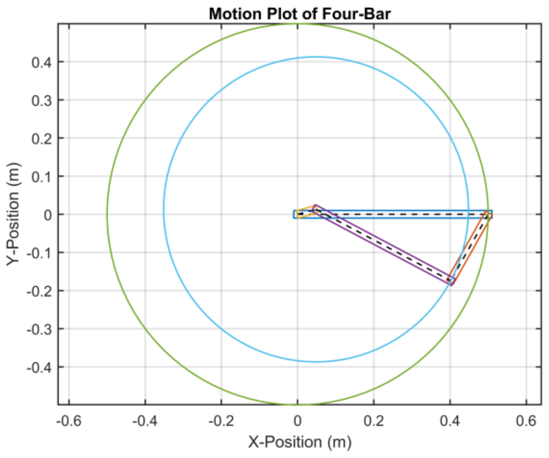


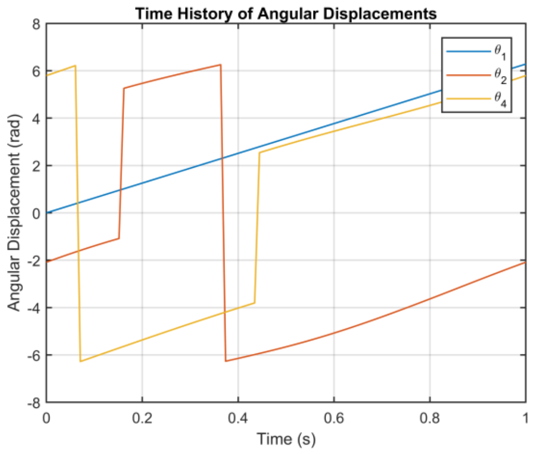


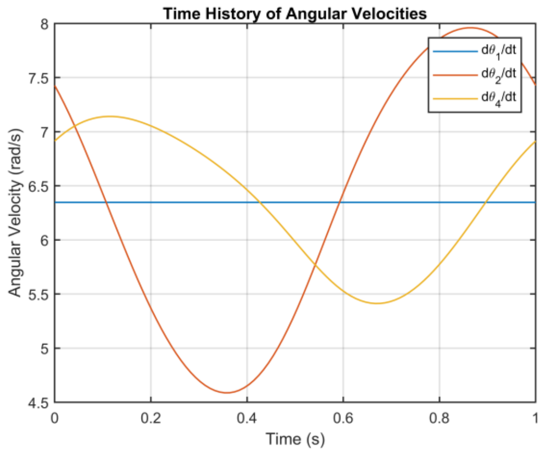


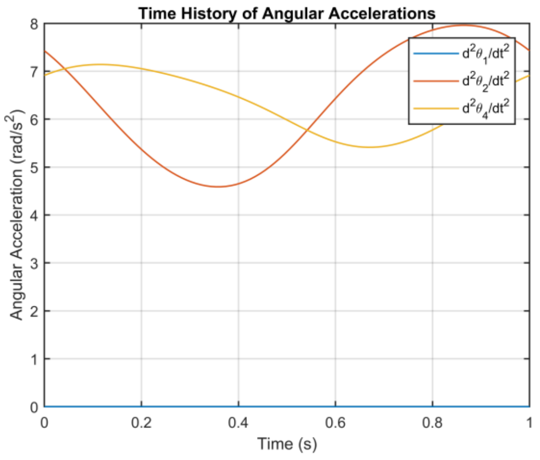


The following four graphs show the motion, angular displacement, angular velocity, and angular acceleration plots of a Crank-Crank Mechanism that has lengths of 0.5, 0.2, 0.05, and 0.4 units, with the 3rd link (ground link) being the shortest.







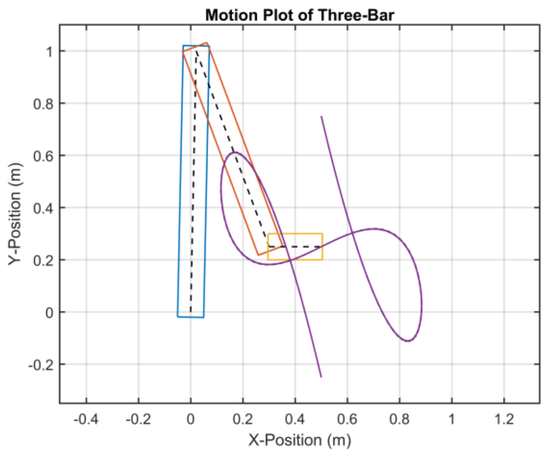


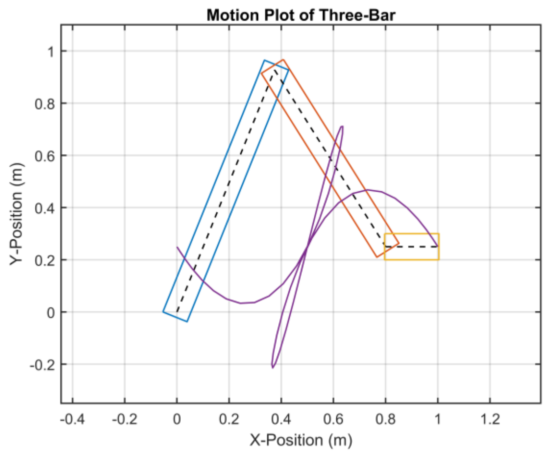
3.2 3R Manipulator

The main_3R file is the base file where the parameters are set and calls the supporting files such as GrashofSanity, Link, mechanisms, and pathPlan. The first portion of the main_3R file sets the parameters of the 3R mechanism. The user will define the lengths of all three links. Then the user will define the geometry parameters of the link overlay. The user will have the option to turn off the overlay later in the code. The next section of the main main_3R file defines the simulation settings such as simulation time step, number of time steps, angle step for circle, and a time vector. Also, during this section the desired end effector position and closure of the mechanism is defined with closure set to 1 being the first closure and closure set to 2 begin the second closure of the mechanism. The next section, Object Construction, will generate the links of the 3R mechanism by calling the supporting files. Firstly, the property of all three links are sent to the supporting file *Link.* Link.m is a class file that accepts length, angle, initial x coordinate, and initial y coordinate. Then stores the instantaneous position of the initial and final coordinates of the vector, the orientation of the vector, velocity and acceleration data, and overlay data. Continuing with Object Construction section of the main_3R file, now that the links have been defined the overlay can be set to true meaning on or false meaning off. If the overlay is set to true, then the overlay shape defined early in the main four-bar file is scaled to the length of each link individually. Next in the Object Construction is the code tells mechanism we want to solve a crank-slider by passing the type crank-slider1. Mechanism.m class accepts type, closure, known lengths, and known angles. Then solves for the unknowns of the mechanism. Since crank-rocker and crank-crank have different unknowns than a rocker-rocker mechanism the Mechanism.m class compares the Type determined in GrashofSanity.m class to if statements to solve for the correct unknowns. Both closures are solved for instantaneously in the Mechanism.m class. Lastly in the object construction section of main four-bar file the code will create a structure P of type *pathPlan* and pass the support file *pathPlan* all the links created, mechanism, closure, and any points of interest. pathPlan.m class accepts links, mechanism, mechtype, closure, and desired position of interest points. Then defines the closure in mechanism to the closure passed to it and initiates the c.update function. The c.update function in *pathPlan* compares the mechtype to a set of if statements to determine whether the mechanism is a four-bar or a 3R manipulator. The function c.update will then send mechanism the link lengths and angles, and solve for the unknown variables. Next, depending the mechanism type the function c.update will construct the mechanism by defining the initial points and angle of each link vector. If the type does not match any of the if statements the following message will print in the command window "The MECHANISM assigned has not been built in this solver," where MECHANISM is the type of mechanism defined by GrashofSanity.m class. Next section of main_3R file, Solution section, is the main loop that plots and solves for all positions as the driving link is rotated at the defined velocity. The solution section is a for loop that runs till number of time steps defined early is achieved. The for loop will update the driving link's angle of structure P for each time step. Next the for loop will store and plot the data of all the links. Finally, the for loop will plot the free join which is the free travel points not attached to the ground link.

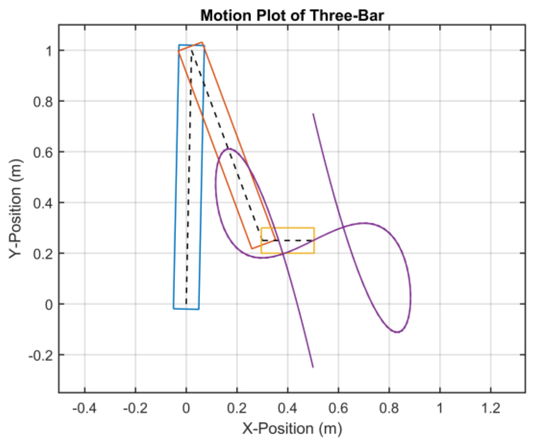
3.2.1 Adaptability in Code

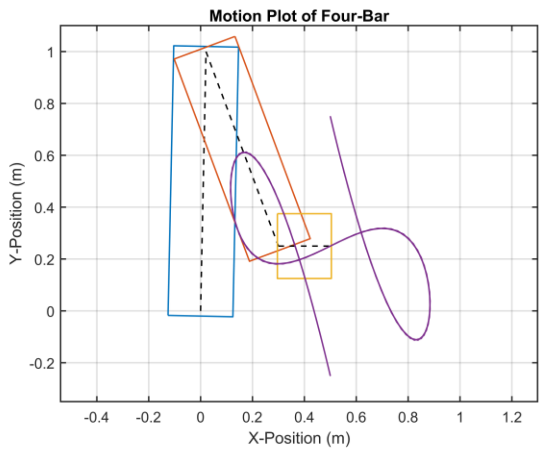
The desired end effector position XR and YR can be changed to any configuration. The following plots show the default configuration on the left and new configuration on the right.



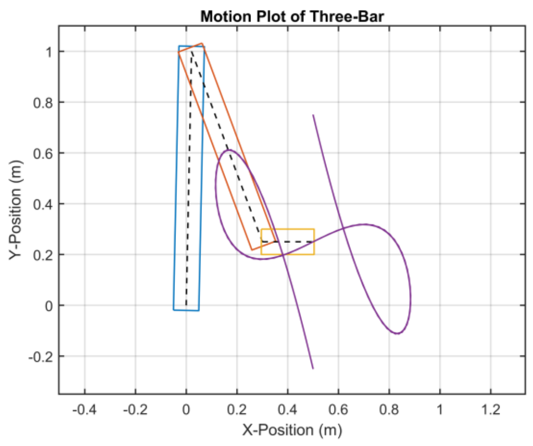


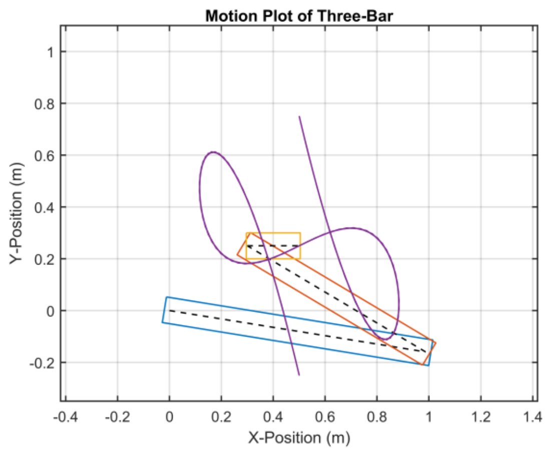
For all plottable configurations the link shape will change if the width of the link (Wid) is changed and/or box_x and box_y local link is changed. The following plots show the default link shape and the width of the link increased.



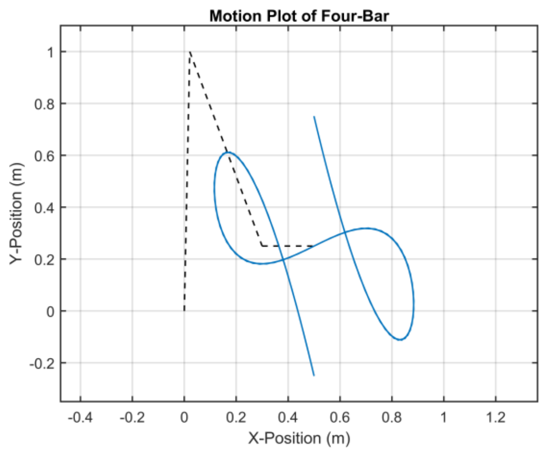


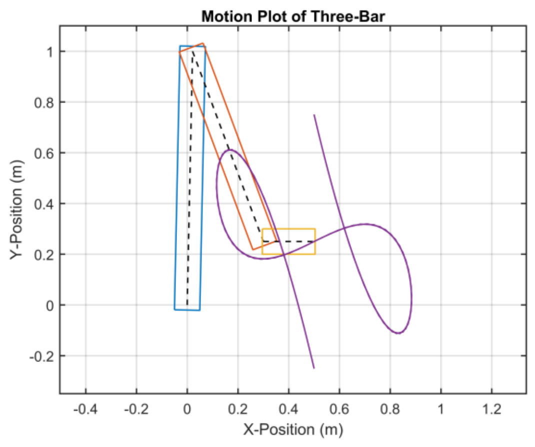
For all plottable configurations both closures are plottable. The following plots show closure 1 displayed on the left and closure two displayed on the right.





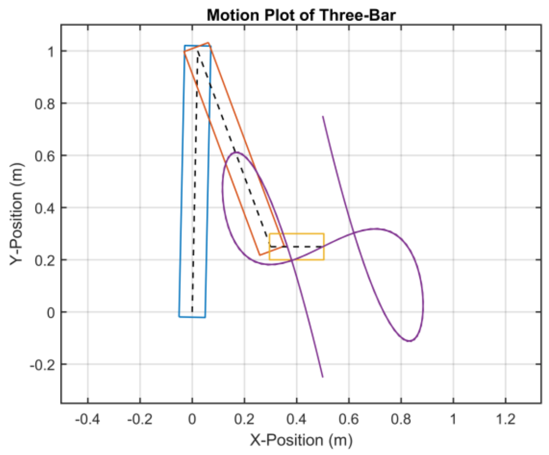
For all plottable configurations the overlay can be toggled on and off by setting overlay equal to true for on, and false for off. The following plots show the overlay set to false on the left and the overlay set to true on the right.

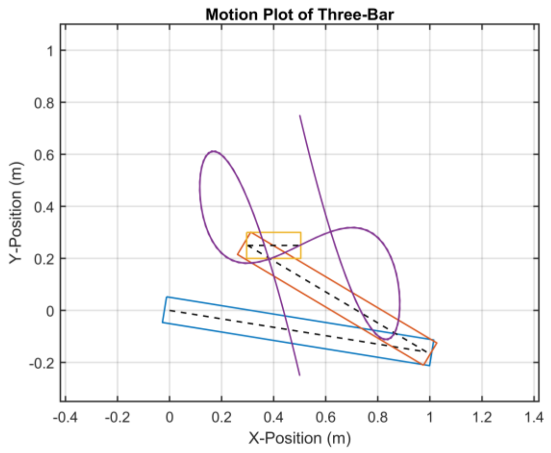




3.2.2 Figures for all Plottable Configurations

The following graphs show the motion plots of a 3R manipulator that has lengths of 1, 0.8, and 0.2 units, with the desired end effector positions set to $XR = rcos(t)\sin(2t) + x_0$ and $YR = rsin(t)\cos(2t) + y_0$. Closure 1 is displayed on the left and closure 2 is displayed on the right.





4 Conclusion

In the end, the results came out as expected. The team created a concise analysis on the four-bar mechanism and the planar 3R manipulator. Both closures for each of the four-bar mechanisms and the 3R manipulator were made. The skeleton code provided was completed and made to function to its maximum performance, therefore a full understanding of object-oriented programming was accomplished. The position, velocity, and acceleration plots were created with the finished code as well as the position plot for the 3R manipulator. Since, the code needed the creation of vector loops to solve for the unknowns the team manually solved for unknowns using the half tangent method. When put together the manual solutions and the code helped the team reach the complete analytic solutions for the four-bar and the 3R manipulator.