SCHOOL OF MECHATRONIC SYSTEMS ENGINEERING
SIMON FRASER UNIVERSITY

MSE 211: Computational Methods for Engineers

Final Project

Group number 07

Wai In Neng, 301270035, signature_____

Ramita Trangkanukulkij, 301270148, signature_____

Ehinmowo Ifeoluwa, 301258850, signature_____

Abstract

This project focuses on the possible uses of four bar link mechanism and how to implement them to create simple designs that can solve problems. It requires us to create our own design and construct a matching physical mechanism. The mechanism we have designed for this project moves items from the storage position of a plane to the ground. After creating the design, we are required to analyze it. These analyses come in the form of Kinematic Analysis, which includes individual displacement, Velocity and Acceleration analysis of the center of mass of each link, dynamic analysis and coupler curve analysis. These various analyses are all carried out using different methods so we can also observe different results that can be obtained from analyzing using different methods.

Table of Contents

Lists of Figures

List of Tables

1. Introduction

Our design involves the use of a four-bar mechanism to unload suitcases, boxes and other forms of luggage from the plane's luggage area to the ground with as little rotation of the items as possible. In order to achieve this design, we decided to use the motion generation algorithm with four specified precision points. We believe that such as mechanism will be useful as it would be easy to build, mount and implement while reducing all the physical labor that is gone through when unloading bags from planes. Creating the mechanism involved first creating a simulation using MATLAB, when creating the simulation arbitrary points where assigned to the provided motion generation function in order to narrow down the right values. After finding the precision points, the relative precision points also had to be found, this was done using the identified precision point and angle of rotation. After the needed values were found a graph depicting the motion of the mechanism through the points was created.

2. Description of the Task

Our group decided to create a four-bar mechanism that will help loading the suitcase from the plane's cabin to the ground with less rotation of motion possible. Therefore, we decide to use motion generation algorithm with the four precision points shown below:

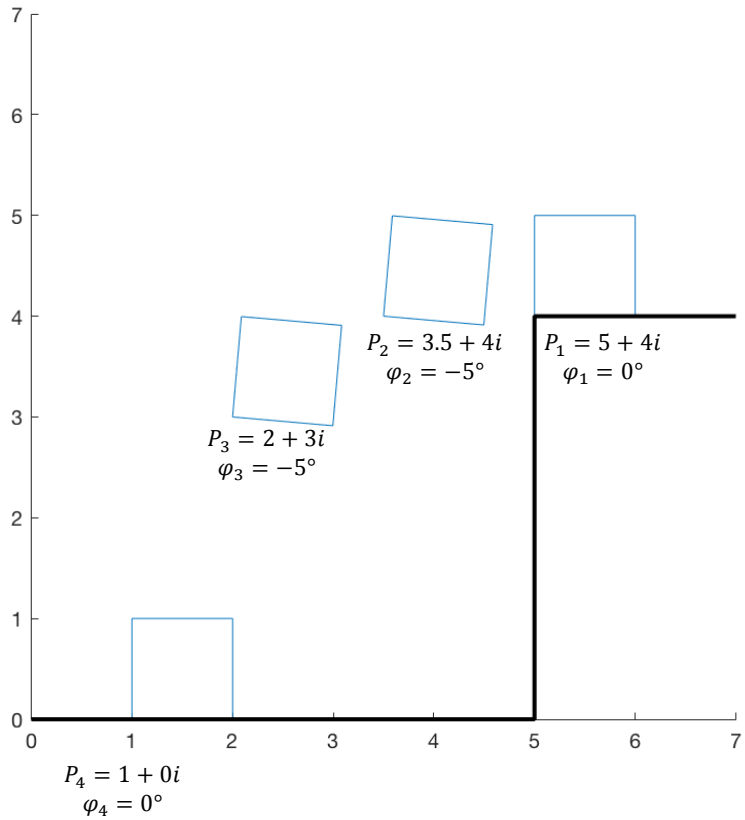


Figure 1: Identified Four Precision Points and Angle

3. Design of Mechanism

In order to define optimization problem for motion generation, the relative precision point should be evaluated using identified precision point and angle of rotation as following:

$$\delta_1 = P_2 - P_1 \qquad \alpha_1 = \varphi_2 - \varphi_1$$

- P: The position of each precision point
- δ: The relative position vector
- φ: The angle of box due to horizontal
- α: The relative rotation

Left side analysis:

The equations that are used are displayed below:

$$f_{jx} = r_{2x} + r'_{3x} + \delta_{jx} - r'_{3x}\cos(\alpha_j) + r'_{3x}\sin(\alpha_j) - r_{2x}\cos(\beta_j) + r_{2y}\sin(\beta_j) = 0$$

$$f_{jy} = r_{2y} + r'_{3y} + \delta_{jy} - r'_{3y}\cos(\alpha_j) - r'_{3x}\sin(\alpha_j) - r_{2y}\cos(\beta_j) - r_{2x}\sin(\beta_j) = 0$$

$$F = \sqrt{(f_{1x})^2 + (f_{1y})^2 + (f_{2x})^2 + (f_{2y})^2 + (f_{3x})^2 + (f_{3y})^2}$$

Right side analysis:

The equations that are used are displayed below:

$$g_{jx} = r_{4x} + r''_{3x} + \delta_{jx} - r''_{3x}\cos(\alpha_j) + r''_{3y}\sin(\alpha_j) - r_{4x}\cos(\gamma_j) + r_{4y}\sin(\gamma_j) = 0$$

$$g_{jy} = r_{4y} + r''_{3y} + \delta_{jy} - r''_{3y}\cos(\alpha_j) - r''_{3x}\sin(\alpha_j) - r_{4y}\cos(\gamma_j) - r_{4x}\sin(\gamma_j) = 0$$

$$G = \sqrt{(g_{1x})^2 + (g_{1y})^2 + (g_{2x})^2 + (g_{2y})^2 + (g_{3x})^2 + (g_{3y})^2}$$

As a result, there are seven unknowns for each side of analysis as shown below that needed to be determined.

Left side analysis		Right side analysis	
r_{2x}	r_{2y}	r_{4x}	r_{4y}
r'_{3x}	r'_{3y}	r''_{3x}	r''_{3y}
β_j, $where\ j = 1,2,3$		γ_j, $where\ j = 1,2,3$	

Table 1: Initial Guess Parameters

Set of Critical Initial Guesses

The following tables represent the set of initial guess that display critical change of the mechanism. There are more initial values that are guessed, but they did not have a great change from the result shown below.

Guessing Value	Results

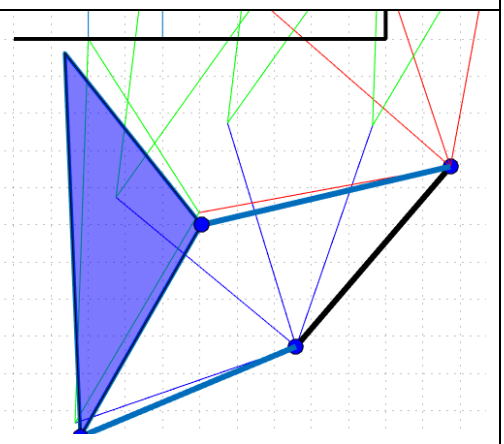
%Optimization Left Side
%Initial values (INPUT INITIAL ESTIMATES)
r2x = 5.58;
r2y = 3.42;
r3ax = 5.5;
r3ay = 2.5;
beta(1)= 10*pi/180; %In radians
beta(2)= -300*pi/180; %In radians
beta(3)= -250*pi/180; %In radians
%Optimization Right side
%Initial values (INPUT INITIAL ESTIMATES)
r4x = 0.6;
r4y = 4.4;
r3bx = 2.5;
r3by = 0.5;
gamma(1)= 25*pi/180; %In radians
gamma(2)= 50*pi/180; %In radians
gamma(3)= -200*pi/180; %In radians
	
	The mechanism cannot follow all precision points and it becomes not feasible after the angle of rotation exceed 180°

Table 2: Guessing Value and Result (First Iteration)

Guessing Value	Results

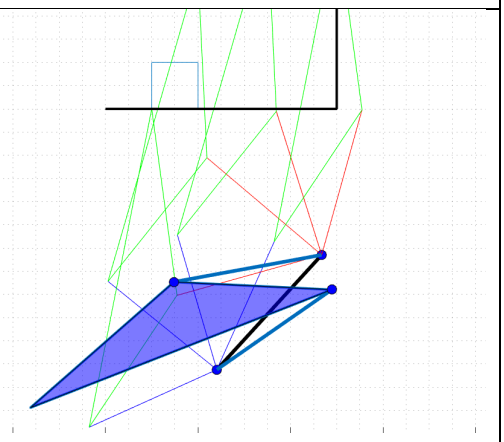
%Optimization Left Side
%Initial values (INPUT INITIAL ESTIMATES)
r2x = 5.58;
r2y = 5.42;
r3ax = 5.5;
r3ay = 5.5;
beta(1)= 10*pi/180; %In radians
beta(2)= -250*pi/180; %In radians
beta(3)= -250*pi/180; %In radians
%Optimization Right side
%Initial values (INPUT INITIAL ESTIMATES)
r4x = 0.6;
r4y = 4.4;
r3bx = 2.5;
r3by = 2.5;
gamma(1)= 25*pi/180; %In radians
gamma(2)= 50*pi/180; %In radians
gamma(3)= -200*pi/180; %In radians
	
	The mechanism still cannot follow all precision points, but it can rotate 360°

Table 3: Guessing Value and Result (Second Iteration)

Guessing Value	Results

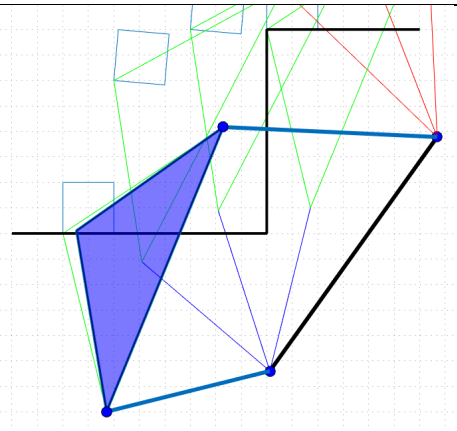
%Optimization Left Side
%Initial values (INPUT INITIAL ESTIMATES)
r2x = 5.58;
r2y = 5.42;
r3ax = 5.5;
r3ay = 5.5;
beta(1)= 10*pi/180; %In radians
beta(2)= -250*pi/180; %In radians
beta(3)= -250*pi/180; %In radians
%Optimization Right side
%Initial values (INPUT INITIAL ESTIMATES)
r4x = 0.6;
r4y = 4.4;
r3bx = 2.5;
r3by = -2.5;
gamma(1)= 25*pi/180; %In radians
gamma(2)= 50*pi/180; %In radians
gamma(3)= -200*pi/180; %In radians
	
	The mechanism can rotate 360°, but it can not follow the precision point P_3 and P_4

Table 4: Guessing Value and Result (Third Iteration)

Guessing Value	Results

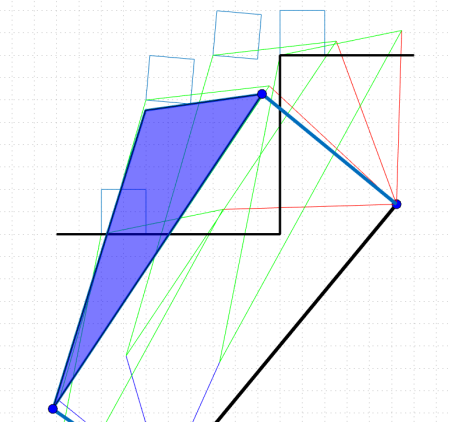
%Optimization Left Side
%Initial values (INPUT INITIAL ESTIMATES)
r2x = 5.58;
r2y = 5.42;
r3ax = 5.5;
r3ay = 5.5;
beta(1)= 10*pi/180; %In radians
beta(2)= -250*pi/180; %In radians
beta(3)= -250*pi/180; %In radians
%Optimization Right side
%Initial values (INPUT INITIAL ESTIMATES)
r4x = 0.6;
r4y = 4.4;
r3bx = 2.5;
r3by = -2.5;
gamma(1)= 25*pi/180; %In radians
gamma(2)= 50*pi/180; %In radians
gamma(3)= -200*pi/180; %In radians
	
	The mechanism can rotate 360°, and it can meet all precision point expect P_3

Table 5: Guessing Value and Result (Fourth Iteration)

After a large number of iterations of trial and error, the appropriate initial guesses and graphical result of feasible mechanism are obtained as illustrated in the following page.

```
%Optimization Left Side
%Initial values (INPUT INITIAL ESTIMATES)
r2x = 0.58;
r2y = 3.42;
r3ax = 2.5;
r3ay = 2.5;
beta(1)= 10*pi/180;  %In radians
beta(2)= -250*pi/180;  %In radians
beta(3)= -250*pi/180;  %In radians
%Optimization Right side
%Initial values (INPUT INITIAL ESTIMATES)
r4x = 0.6;
r4y = 4.4;
r3bx = -2.5;
r3by = 0.5;
gamma(1)= 25*pi/180;  %In radians
gamma(2)= 50*pi/180;  %In radians
gamma(3)= -200*pi/180;  %In radians
```

Table 6: Appropriate Initial Values

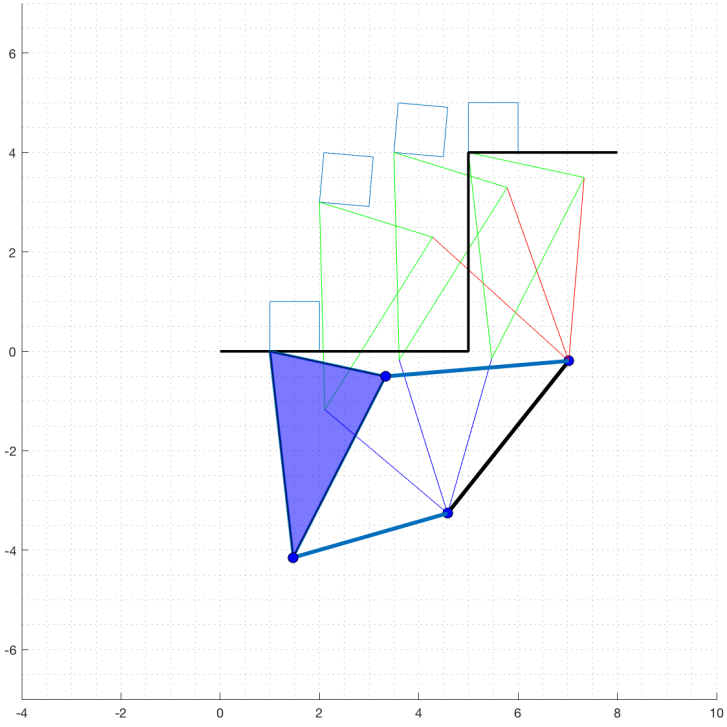


Figure 2: Resulting Mechanism

The following table represented the solutions obtained with optimization that are the length and angle of each link.

LFval	RFval	r1	r2	r3	r4
0.00000596	0.00000460	3.91882993	3.23490430	4.09244415	3.70202612
beta1	beta2	beta3	gamma1	gamma2	gamma3
33.47858461	65.83725805	121.93538690	384.41769000	412.60236923	99.63996916

Table 7: Obtained Solutions with Optimization

Prototype

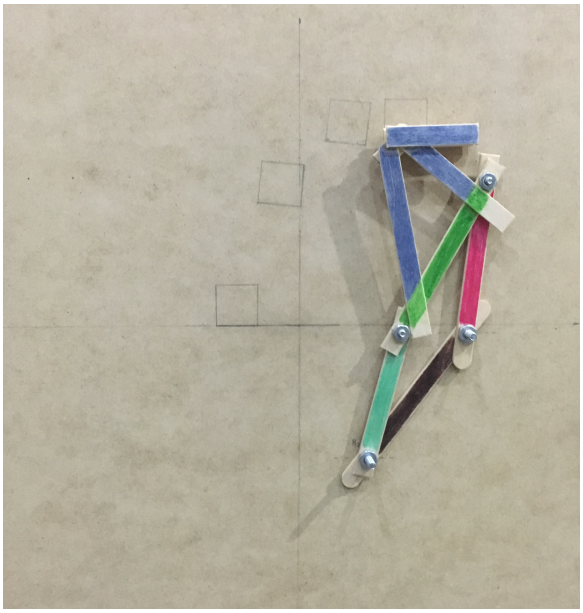


Figure 3: Prototype at Precision Point 1

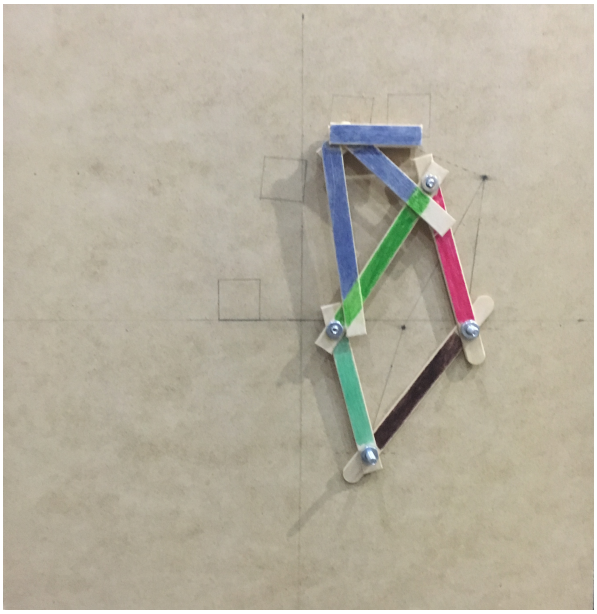


Figure 4: Prototype at Precision Point 2

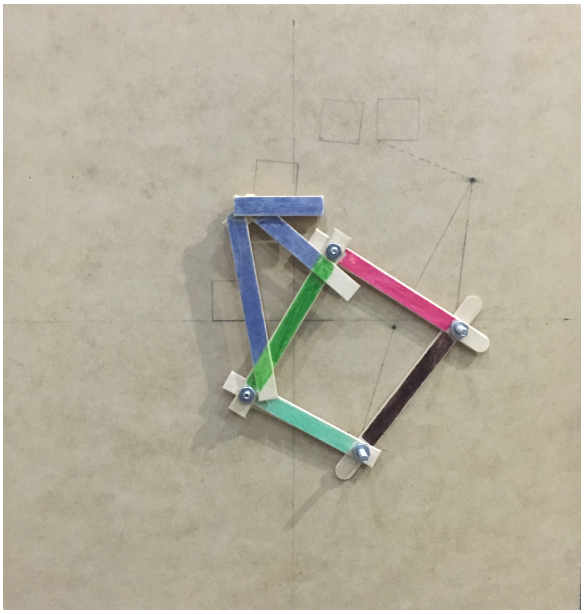


Figure 5: Prototype at Precision Point 3

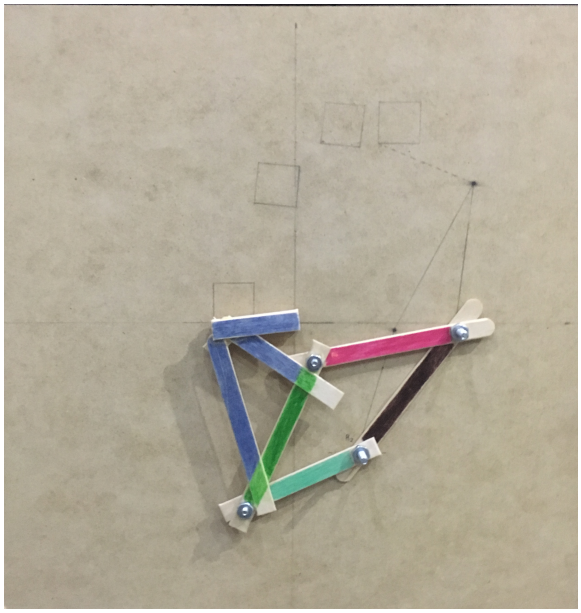


Figure 6: Prototype at Precision Point 4

4. Kinematic Analysis

4.1 Displacement Analysis

The displacement of four-bar mechanism can be evaluated by using numerical method.

The vector representation of four-bar mechanism can be written as shown:

$$-\vec{R}_1 + \vec{R}_2 + \vec{R}_3 - \vec{R}_4 = 0$$

The vectors can be decomposed into X and Y component as following:

$$-r_1 + r_2 cos\theta_2 + r_3 cos\theta_3 - r_4 cos\theta_4 = 0$$

$$r_2 sin\theta_2 + r_3 sin\theta_3 - r_4 sin\theta_4 = 0$$

Since the length of each bar (r_1, r_2, r_3, r_4), and input angle (θ_2) are known, the equation above can be re-written as shown below:

$$r_3 cos\theta_3 = a + r_4 cos\theta_4, \qquad where\ a = r_1 - r_2 cos\theta_2$$

$$r_3 sin\theta_3 = b + r_4 sin\theta_4, \qquad where\ b = -r_2 sin\theta_2$$

Two equations above can be reduced by squaring and adding:

$$acos\theta_4 + bsin\theta_4 = c, \qquad where\ c = \frac{r_3^2 - a^2 - b^2 - r_4^2}{2r_4}$$

Therefore, the angular displacement of link R4 (θ_4) can be obtained by using **Newton-Raphson method** with the function shown below:

$$f(x) = acos\theta_4 + bsin\theta_4 - c$$

Once the set of angular displacement of link R4 (θ_4) are found, the set of angular displacement of link R3 (θ_3) can be evaluated by simple substitution:

$$sin\theta_3 = \frac{b + r_4 sin\theta_4}{r_3}$$

$$cos\theta_3 = \frac{a + r_4 cos\theta_4}{r_3}$$

$$\theta_3 = atan2(sin\theta_3, cos\theta_3)$$

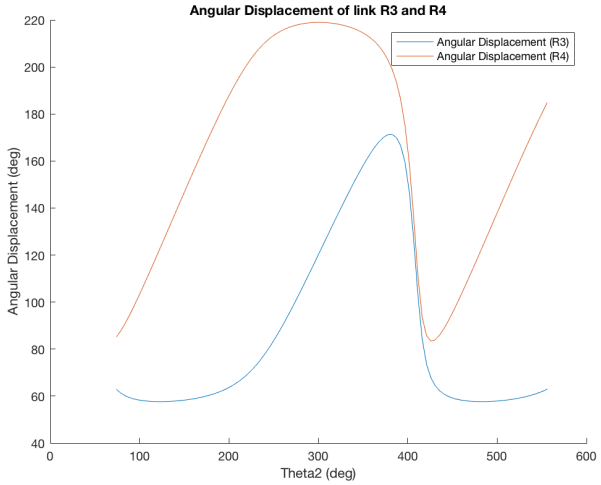


Figure 7: θ_3 and θ_4 vs θ_2

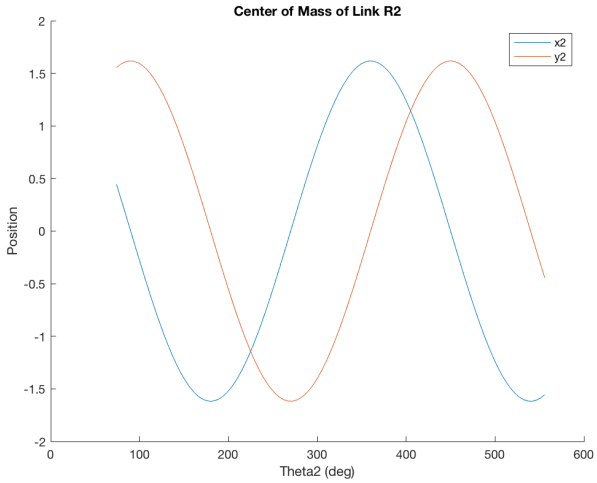


Figure 8: Center of Mass of Link R2 vs θ_2

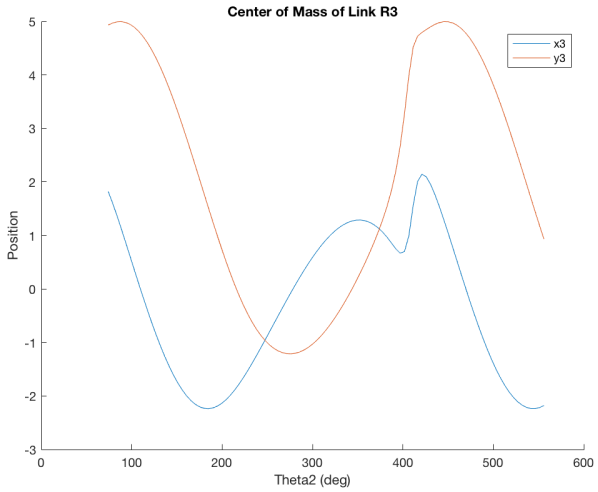


Figure 9: Center of Mass of Link R3 vs θ_2

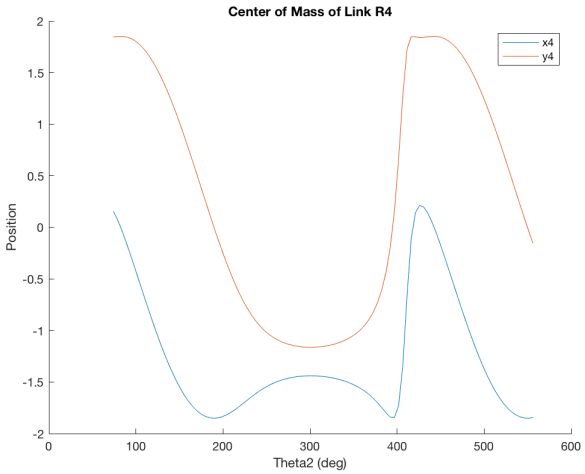


Figure 10: Center of Mass of Link R4 vs θ_2

4.2 Velocity and Acceleration Analysis

Analytic Solution

To compute angular velocity and acceleration (Assume $\theta_2 = 1\ rad/s$):

$$\begin{bmatrix} -r_3 sin\theta_3 & r_4 sin\theta_4 \\ r_3 cos\theta_3 & -r_4 cos\theta_4 \end{bmatrix} \begin{bmatrix} \dot{\theta}_3 \\ \dot{\theta}_4 \end{bmatrix} = \begin{bmatrix} r_2 sin\theta_2 \dot{\theta}_2 \\ -r_2 cos\theta_2 \dot{\theta}_2 \end{bmatrix}$$

$$\begin{bmatrix} -r_3 sin\theta_3 & r_4 sin\theta_4 \\ r_3 cos\theta_3 & -r_4 cos\theta_4 \end{bmatrix} \begin{bmatrix} \ddot{\theta}_3 \\ \ddot{\theta}_4 \end{bmatrix} = \begin{bmatrix} r_2 sin\theta_2 \ddot{\theta}_2 + r_2 cos\theta_2 \dot{\theta}_2^{\,2} + r_3 cos\theta_3 \dot{\theta}_3^{\,2} - r_4 cos\theta_4 \dot{\theta}_4^{\,2} \\ -r_2 cos\theta_2 \ddot{\theta}_2 + r_2 sin\theta_2 \dot{\theta}_2^{\,2} + r_3 sin\theta_3 \dot{\theta}_3^{\,2} - r_4 sin\theta_4 \dot{\theta}_4^{\,2} \end{bmatrix}$$

To compute linear velocity of each center of mass of each link:

$$V_{2x} = -\frac{r_2}{2}\dot{\theta}_2 \sin\theta_2 \qquad V_{2y} = \frac{r_2}{2}\dot{\theta}_2 \cos\theta_2$$

$$V_{3x} = -r_2\dot{\theta}_2 \sin\theta_2 - \frac{r_3}{2}\dot{\theta}_3 \sin\theta_3 \qquad V_{3y} = r_2\dot{\theta}_2 \cos\theta_2 + \frac{r_3}{2}\dot{\theta}_3 \cos\theta_3$$

$$V_{4x} = -\frac{r_4}{2}\dot{\theta}_4 \sin\theta_4 \qquad V_{4y} = \frac{r_4}{2}\dot{\theta}_4 \cos\theta_4$$

To compute linear acceleration of each center of mass of each link:

$$A_{2x} = -\frac{r_2}{2}\ddot{\theta}_2 \sin\theta_2 - \frac{r_2}{2}\dot{\theta}_2^{\,2} \cos\theta_2 \qquad A_{2y} = \frac{r_2}{2}\ddot{\theta}_2 \cos\theta_2 - \frac{r_2}{2}\dot{\theta}_2^{\,2} \sin\theta_2$$

$$A_{4x} = -\frac{r_4}{2}\ddot{\theta}_4 \sin\theta_4 - \frac{r_4}{2}\dot{\theta}_4^{\,2} \cos\theta_4 \qquad A_{4y} = \frac{r_4}{2}\ddot{\theta}_4 \cos\theta_4 - \frac{r_4}{2}\dot{\theta}_4^{\,2} \sin\theta_4$$

$$A_{3x} = -r_2\ddot{\theta}_2 \sin\theta_2 - r_2\dot{\theta}_2^{\,2} \cos\theta_2 - \frac{r_3}{2}\ddot{\theta}_3 \sin\theta_3 - \frac{r_3}{2}\dot{\theta}_3^{\,2} \cos\theta_3$$

$$A_{3y} = r_2\ddot{\theta}_2 \cos\theta_2 - r_2\dot{\theta}_2^{\,2} \sin\theta_2 + \frac{r_3}{2}\ddot{\theta}_3 \cos\theta_3 - \frac{r_3}{2}\dot{\theta}_3^{\,2} \sin\theta_3$$

The following figures illustrated the obtained both angular and linear velocity and acceleration of each link by using the analytical method.

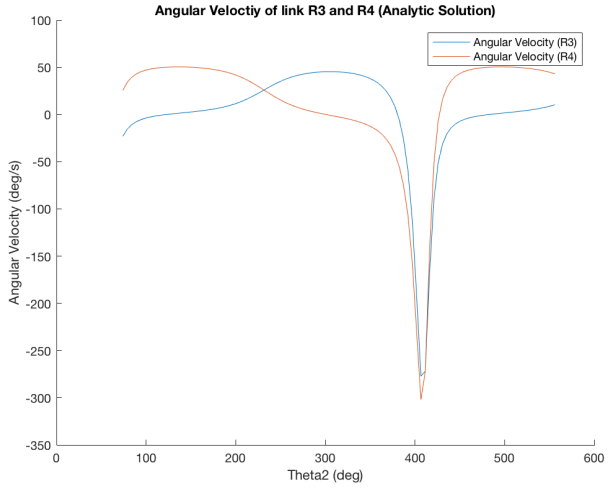


Figure 11: $\dot{\theta}_3$ *and* $\dot{\theta}_4$ using Analytic Solution vs θ_2

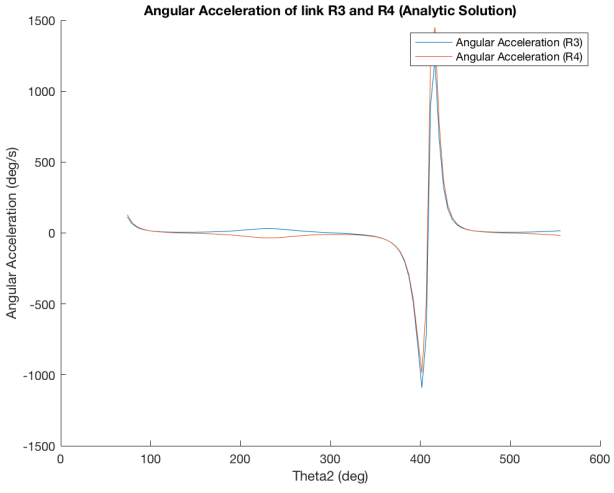


Figure 12: $\ddot{\theta}_3$ *and* $\ddot{\theta}_4$ using Analytic Solution vs θ_2

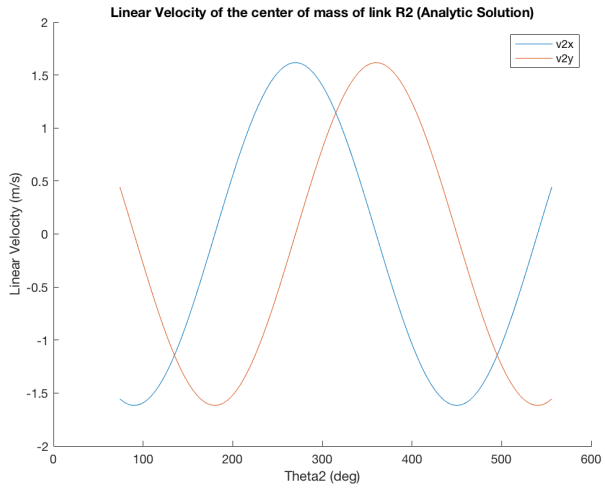


Figure 13: v_2 using Analytic Solution vs θ_2

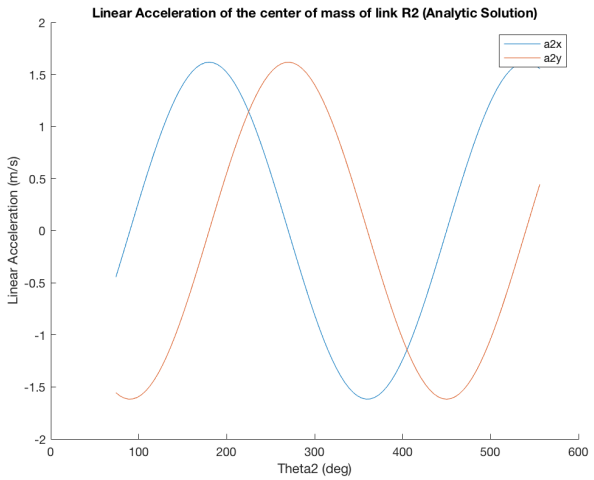


Figure 14: a_2 using Analytic Solution vs θ_2

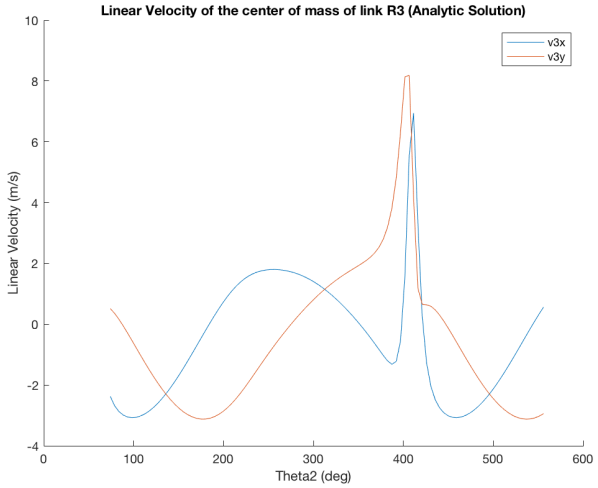


Figure 15: v_3 using Analytic Solution vs θ_2

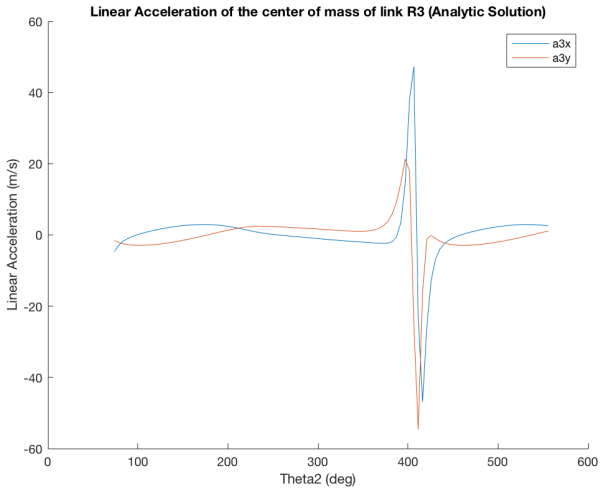


Figure 16: a_3 using Analytic Solution vs θ_2

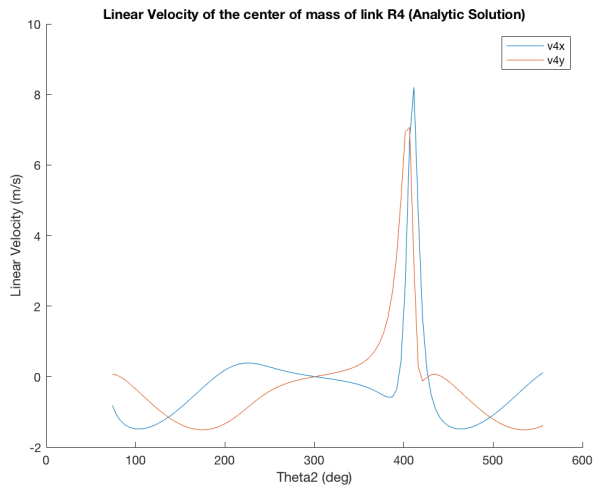


Figure 17: v_4 using Analytic Solution vs θ_2

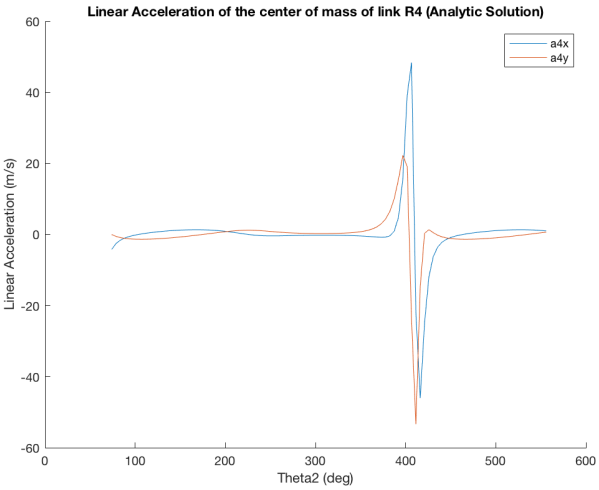


Figure 18: a_4 using Analytic Solution vs θ_2

Numerical Solution

Centered Difference Approximation (Basic)

$$f'(x) = \frac{f(x_{i+1}) - f(x_{i-1})}{2h}, \qquad f''(x) = \frac{f(x_{i+1}) - 2f(x_i) + f(x_{i-1})}{h^2}$$

Centered Difference Approximation (Expanded)

$$f'(x) = \frac{-f(x_{i+2}) + 8f(x_{i+1}) - 8f(x_{i-1}) + f(x_{i-2})}{12h}$$

$$f''(x) = \frac{-f(x_{i+2}) + 16f(x_{i+1}) - 30f(x_{i-1}) + 16f(x_{i-1}) - f(x_{i-2})}{12h^2}$$

The following figures illustrated the obtained both angular and linear velocity and acceleration of each link by using the centered difference approximation.

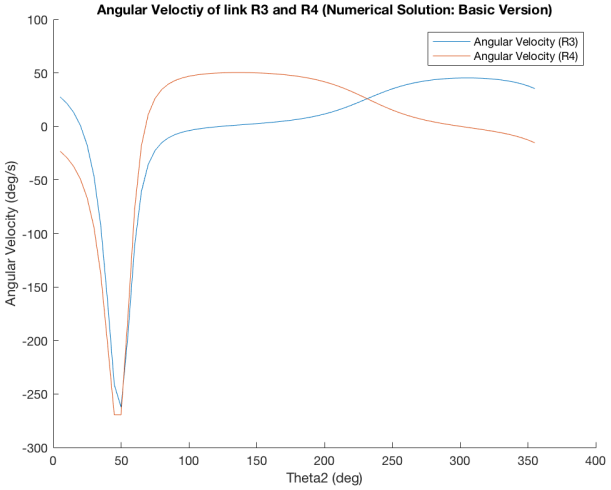


Figure 19: $\dot{\theta}_3$ *and* $\dot{\theta}_4$ (Numerical Solution: Basic)

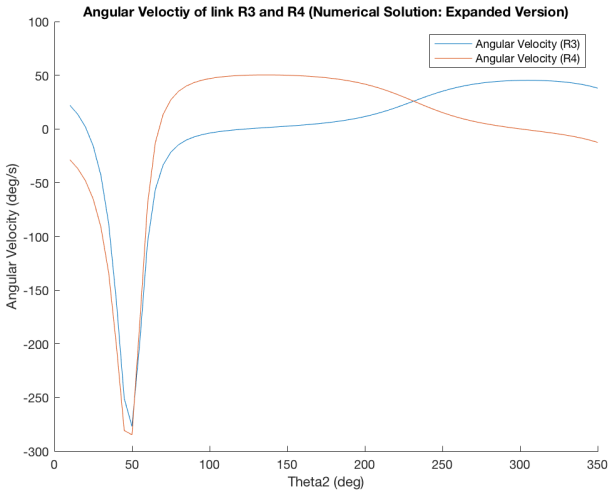


Figure 20: $\dot{\theta}_3$ *and* $\dot{\theta}_4$ (Numerical Solution: Expand)

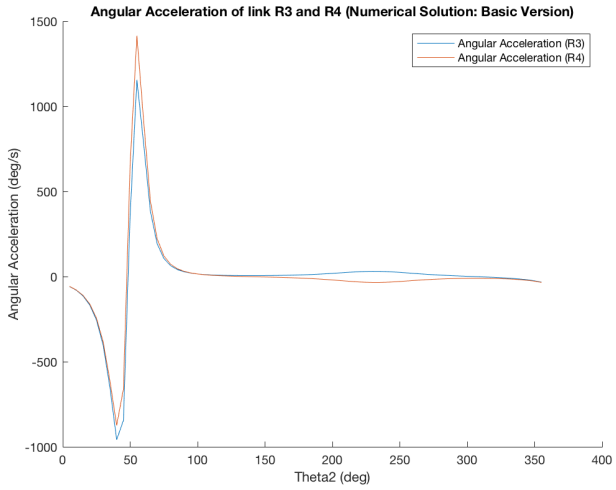


Figure 21: $\ddot{\theta}_3$ *and* $\ddot{\theta}_4$ (Numerical Solution: Basic)

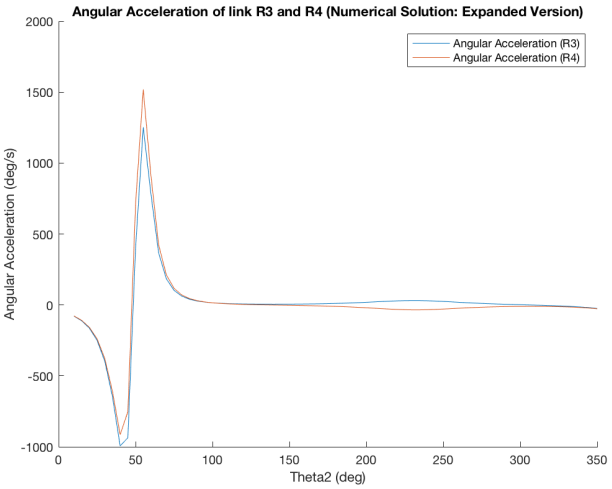


Figure 22: $\ddot{\theta}_3$ *and* $\ddot{\theta}_4$ (Numerical Solution: Expand)

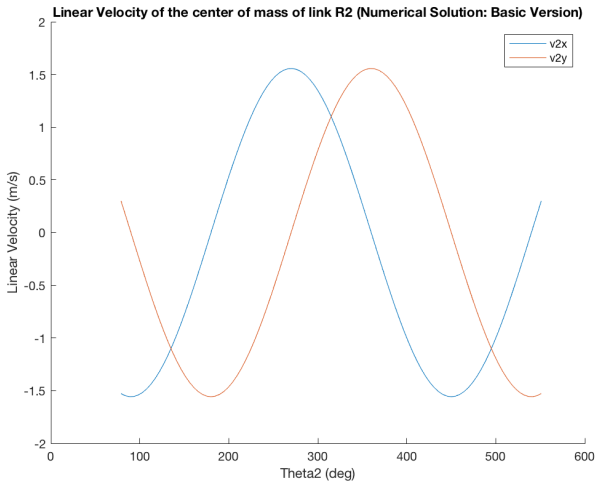


Figure 23: v_2 (Numerical Solution: Basic)

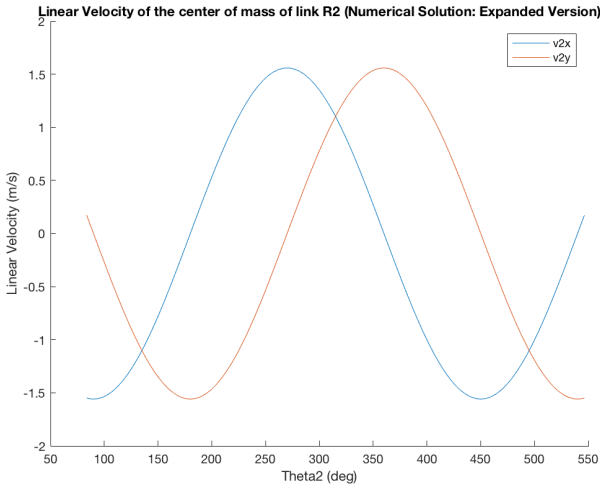


Figure 24: v_2 (Numerical Solution: Expand)

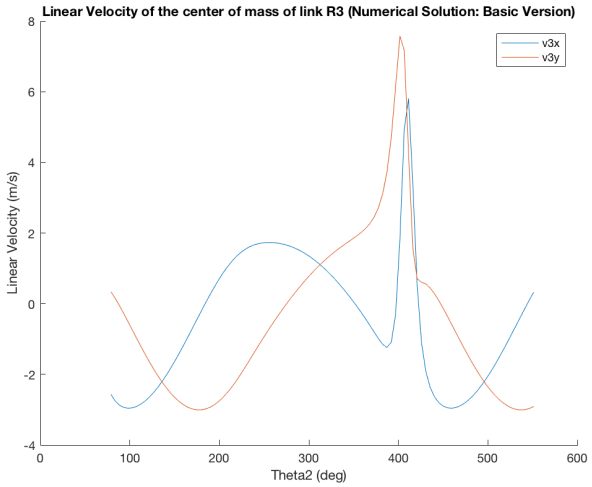


Figure 25: v_3 (Numerical Solution: Basic)

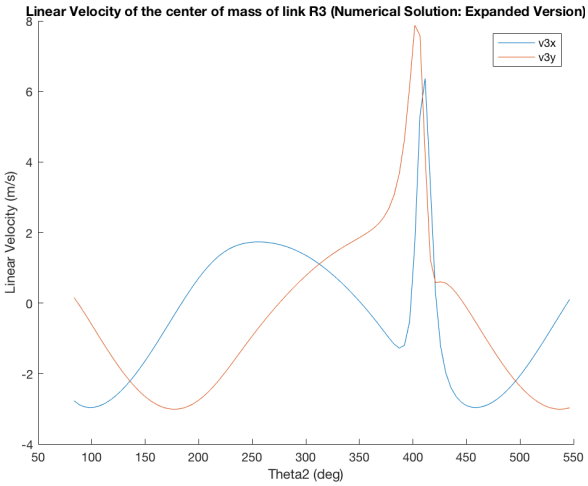


Figure 26: v_3 (Numerical Solution: Expand)

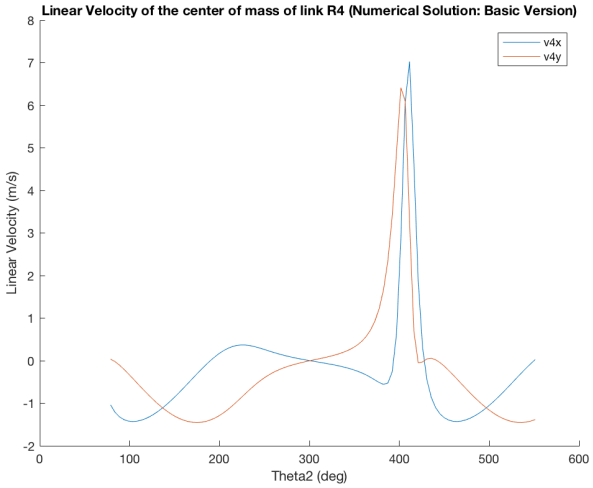


Figure 27: v_4 (Numerical Solution: Basic)

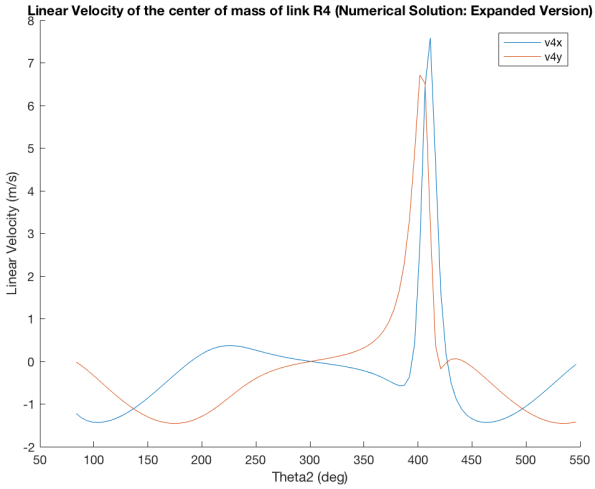


Figure 28: v_4 (Numerical Solution: Expand)

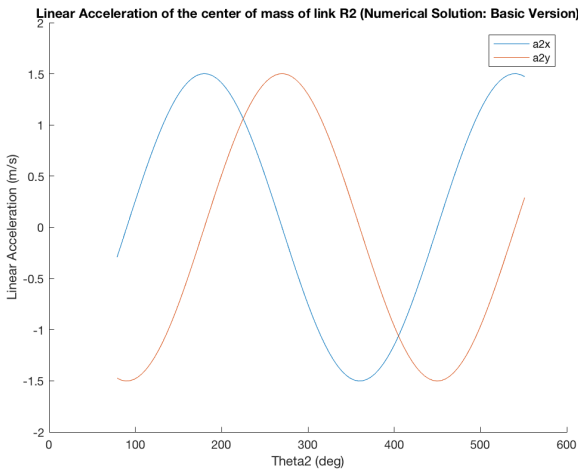


Figure 29: a_2 (Numerical Solution: Basic)

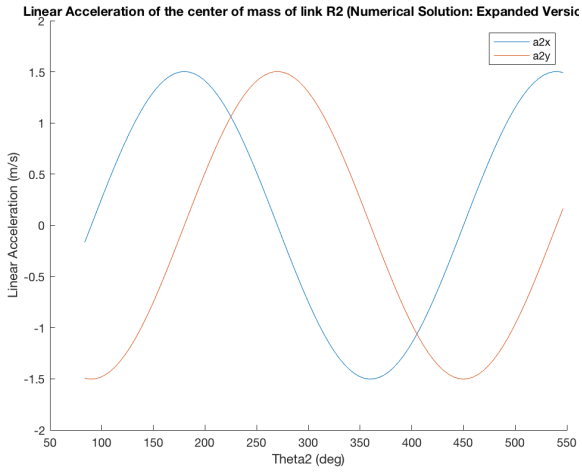


Figure 30: a_2 (Numerical Solution: Expand)

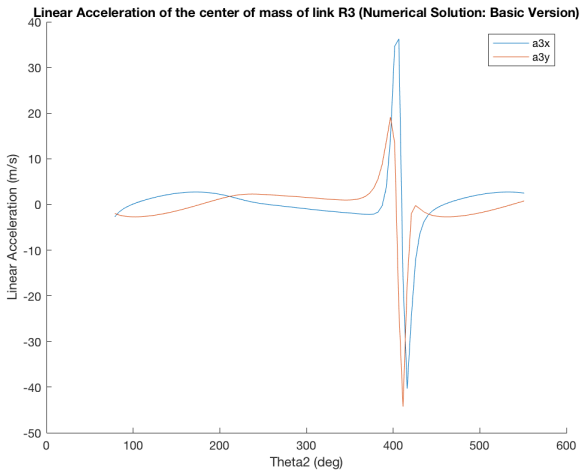


Figure 31: a_3 (Numerical Solution: Basic)

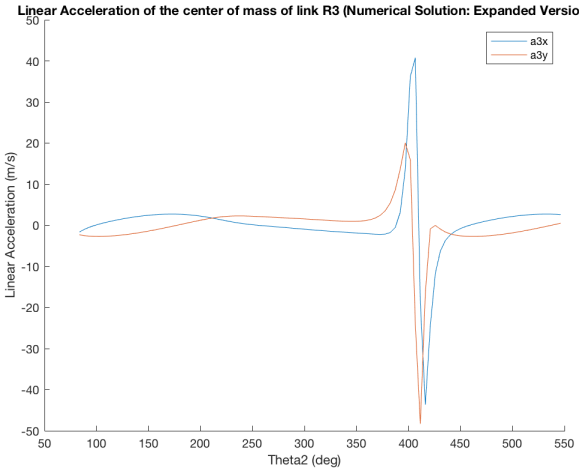


Figure 32: a_3 (Numerical Solution: Expand)

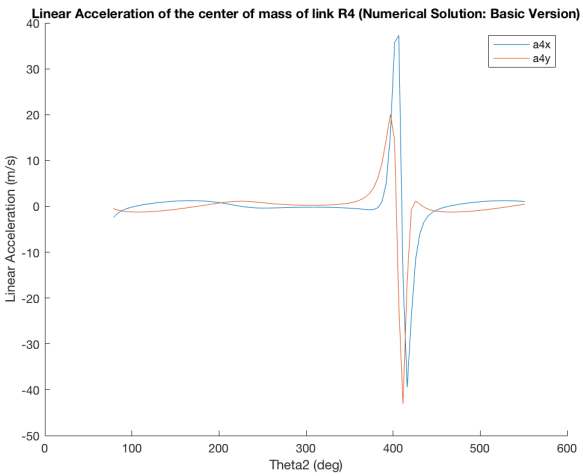


Figure 33: a_4 (Numerical Solution: Basic)

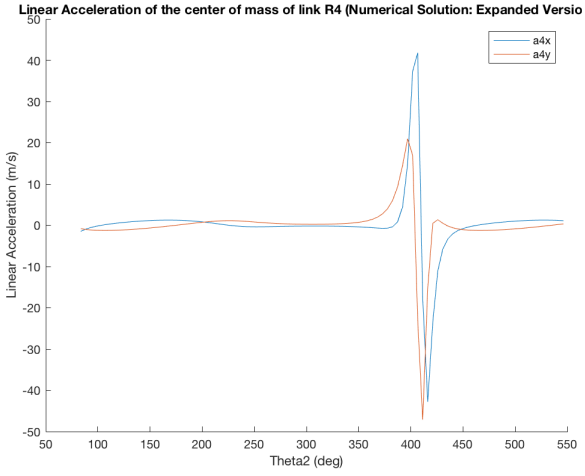


Figure 34: a_4 (Numerical Solution: Expand)

According to results shown above, it can be seen that the obtained graphs by using analytical method are same as the graphs that are obtained by using both basic and expanded form of a center difference approximation.

5. Dynamic Analysis

Disconnect all the members

Link R2

- $F_{32:}$ reaction of 3 on 2
- M_{12}: input torque

$$+\rightarrow \Sigma F_{2x} = m_2 a_{G2x} \Rightarrow F_{32x} + F_{12x} = m_2 a_{G2x}$$

$$+\uparrow \Sigma F_{2y} = m_2 a_{G2y} \Rightarrow F_{32y} + F_{12y} = m_2 a_{G2y}$$

$$+\circlearrowleft \Sigma M_{G2} = I_{G2}\ddot{\theta}_2 = 0 \Rightarrow M_{12} - F_{12y}r_{21x} + F_{12x}r_{21y} + F_{32y}r_{23x} - F_{32x}r_{23y} = 0$$

Link R3

$$\Sigma F_{3x} = m_3 a_{G3x} \Rightarrow F_{23x} + F_{43x} = F_{43x} - F_{32x} = m_3 a_{G3x}$$

$$\Sigma F_{3y} = m_3 a_{G3y} \Rightarrow F_{43y} - F_{32y} = m_3 a_{G3y}$$

$$\Sigma M_{G3} = I_{G3}\ddot{\theta}_3 \Rightarrow -F_{32y}r_{32x} - F_{32x}r_{32y} - F_{43y}r_{34x} - F_{43x}r_{34y} = I_{G3}\ddot{\theta}_3$$

Link R4

$$\Sigma F_{4x} = m_4 a_{G4x} \Rightarrow F_{14x} - F_{43x} = m_4 a_{G4x}$$

$$\Sigma F_{4y} = m_4 a_{G4y} \Rightarrow F_{14y} - F_{43y} = m_4 a_{G4y}$$

$$\Sigma M_{G4} = I_{G4}\ddot{\theta}_4 \Rightarrow -F_{14y}r_{41x} + F_{14x}r_{41y} - F_{43y}r_{43x} + F_{43x}r_{34y} = I_{G4}\ddot{\theta}_4$$

For our prototype, we used Popsicle stick as the material. The specification of one is 1.33g in weight and 11.43cm in length. Therefore, we can find the specification of our links.

Link	r1	r2	r3	r4
Length (cm)	3.919	3.235	4.092	3.702
Mass (g)	0.4560	0.3764	0.4762	0.4308
I (cm2)	0.5836	0.3282	0.6646	0.4920

Table 8: Specification of each Link

$$\begin{array}{cccccccccc} F_{12x} & F_{12y} & F_{32x} & F_{32y} & F_{43x} & F_{43y} & F_{14x} & F_{14y} & M_{12} & \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & \\ r_{21y} & -r_{21x} & -r_{23y} & r_{23x} & 0 & 0 & 0 & 0 & 1 & \\ 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & \\ 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & \\ 0 & 0 & -r_{32y} & -r_{32x} & -r_{34y} & -r_{34x} & 0 & 0 & 0 & \\ 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 & \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 & \\ 0 & 0 & 0 & 0 & r_{43y} & -r_{43x} & r_{41y} & -r_{41x} & 0 & \end{array} \begin{array}{c} \\ F_{12x} \\ F_{12y} \\ F_{32x} \\ F_{32y} \\ F_{43x} \\ F_{43y} \\ F_{14x} \\ F_{14y} \\ M_{12} \end{array} = \begin{array}{c} \\ m_2 a_{G2x} \\ m_2 a_{G2y} \\ 0 \\ m_3 a_{G3x} \\ m_3 a_{G3y} \\ I_{G3}\ddot{\theta}_3 \\ m_4 a_{G4x} \\ m_4 a_{G4y} \\ I_{G4}\ddot{\theta}_4 \end{array}$$

Table 9: System of Equation for Dynamic Analysis

The following figures illustrated the change of torque with respect to the change of angle of link 2.

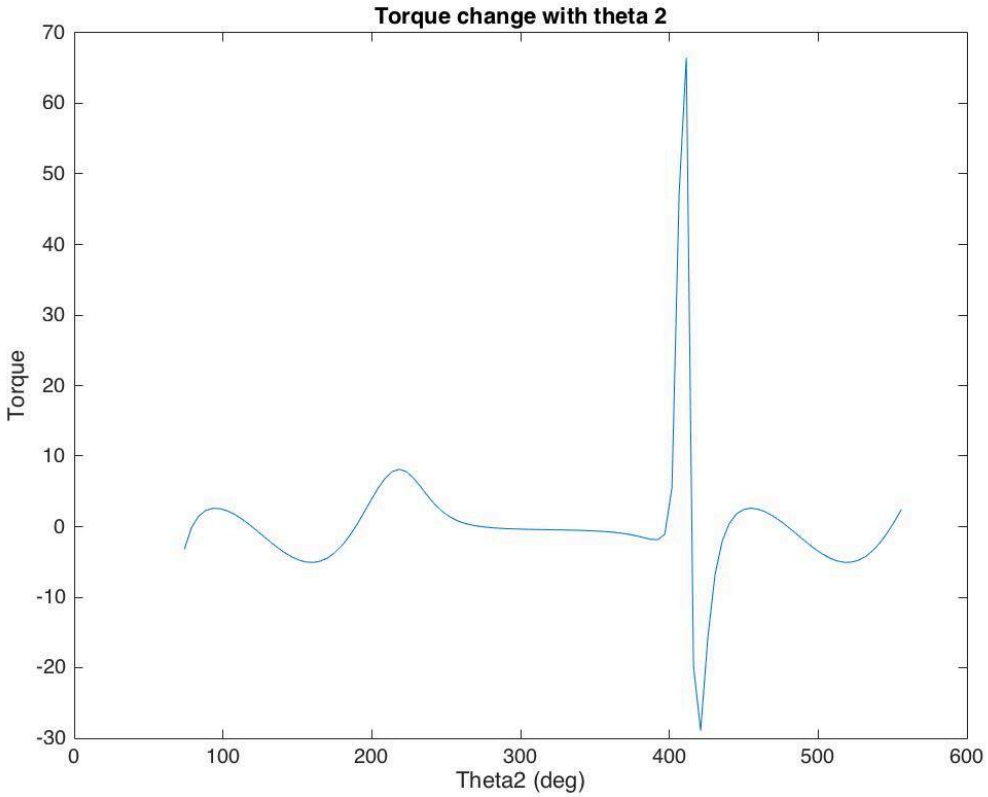


Figure 35: Torque vs θ_2

6. Coupler Curve

Curve Fitting

Linear Regression

For each link, linear regression is found by the following steps. The calculation for link 2 is shown.

i	theta_2	x_i	y_i	$(y_i-y)^2$	$(y-a_0-a_1x)^2$
1	1.2921	0.445	1.555	0.8423	0.2123
2	1.5444	0.0426	1.6169	0.9597	0.0000
3	1.7968	-0.3624	1.5763	0.8818	0.1259
4	2.0491	-0.7445	1.4369	0.6395	0.3498
5	2.3015	-1.0794	1.2046	0.3219	0.4742
6	2.5538	-1.3460	0.8969	0.0674	0.4137
7	2.8062	-1.5273	0.5324	0.0110	0.2089
8	3.0585	-1.6119	0.1343	0.2529	0.0202
9	3.3108	-1.5943	-0.2724	0.8274	0.0794
10	3.5632	-1.4758	-0.6619	1.6878	0.6208
11	3.8155	-1.2638	-1.0094	2.7114	1.8062
Sum		-10.5178	7.0096	9.2032	4.3115

n=11

$$\bar{x} = \frac{\sum x_i}{n} = \frac{-10.52}{11} = -0.9562$$

$$\bar{y} = \frac{\sum y_i}{n} = \frac{7.0096}{11} = 0.6372$$

$$\sum {x_i}^2 = 15.11 \qquad \sum x_i y_i = -1.7305$$

$$a_1 = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{n\sum {x_i}^2 - (\sum x_i)^2} = \frac{11(-1.7305) - (-0.9562)(0.6372)}{11(15.11) - (-0.9562)^2} = 0.98387184$$

$$a_0 = \bar{y} - a_1\bar{x} = 0.6372 - (0.9839)(-0.9562) = 1.57797884$$

$$y = a_0 + a_1 x = 1.578 + 0.98391x$$

$$Linear\ Regression: y = 1.578 + 0.9839x$$

Polynomial

For the quadratic, cubic and quartic polynomial regression, we used Matlab function polyfit to find the coefficients. For example, quadratic polynomial regression is

```
p2 = polyfit(x,y,2);
```

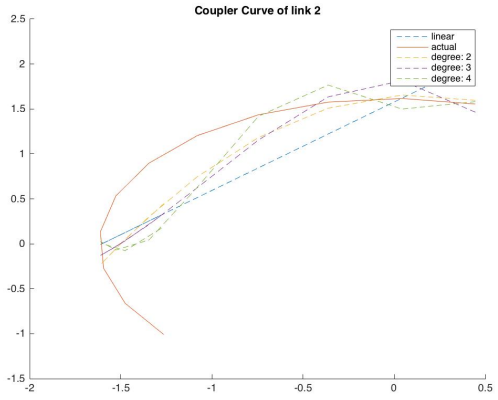


Figure 36: Interpolation of Link 2

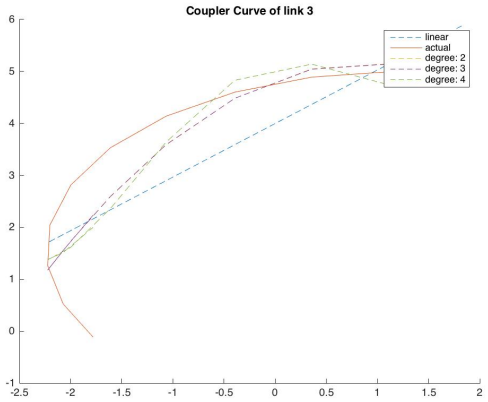


Figure 37: Interpolation of Link 3

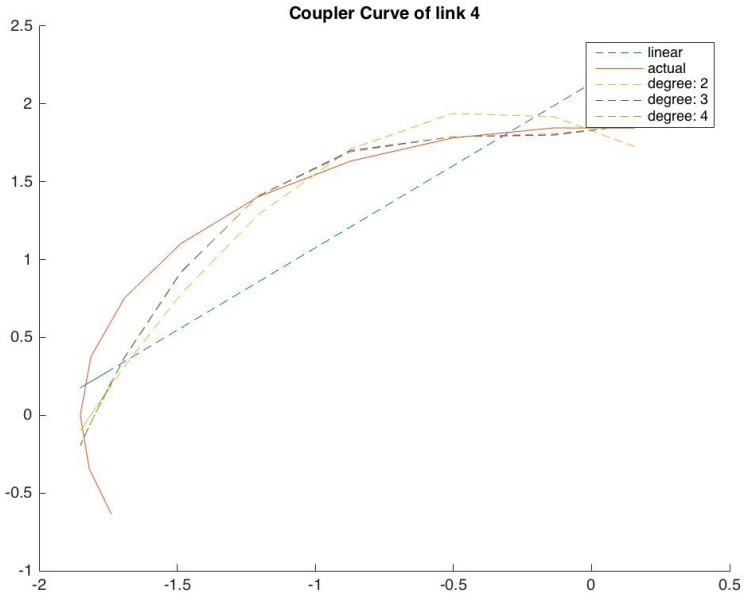


Figure 38: Interpolation of Link 4

By the results shown above, since the coupler curves do not follow a straight line, linear regression modelling does not fit any of the link. On the other hand, 2nd degree polyfit fits the three links, which look like the fact that the coupler curve follows parabola.

Interpolation

In this part, we used Newton's divided difference to interpolate the extracted points of the couple curve. From the previous results, we know that 2nd degree polynomial regression fits the curves, so 2nd degree Newton's interpolation can give us high accuracy.

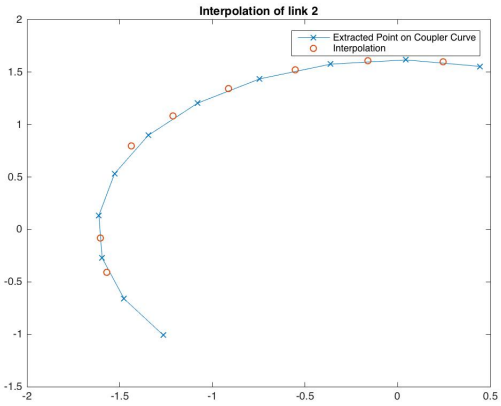


Figure 39: Interpolation of Link 2

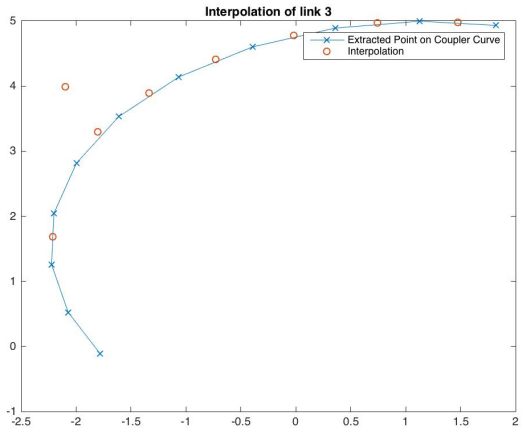


Figure 40: Interpolation of Link 3

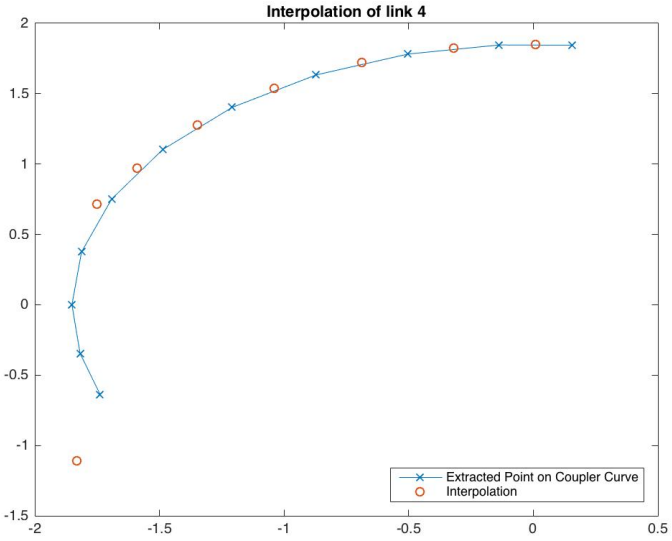


Figure 41: Interpolation of Link 4

From the plots, the interpolation of link 2 looks accurate, and the interpolated points are on coupler curve.

7. Conclusion

The purpose of this project was to learn how to design a four-bar link mechanism using either motion or path generation. This mechanism is to be designed in such a way so as to be able to perform a specified function. The mechanism we have designed for this project serves the purpose of being used to move items from an elevated position to ground level with the instance used in the project referring to removing baggage from planes. In order to create the design, we needed to first optimize the design; this was done using the motion generation function. Initially when using this function the precision points were unknown; hence they had to be guessed to obtain the right values. After the optimization, the kinematic analysis was then performed. The kinematic analysis was divided into displacement, velocity and acceleration analysis of the center of mass of each link. These are analyses were performed using both analytical and numerical methods. The analytical method used was the Newton-Raphson method and the numerical methods used were the centered difference approximation. Both methods were observed to give immensely similar results as expected. Calculating the moment of inertia on each of the links then performed the dynamic analysis; this was calculated using the system of linear equations. This was used to find the magnitude of the force causing the moment in each of the bars. Finally, a coupler curve was used to extract distinct points from the path of the motion, these points were the curve fit was different orders of magnitude while their curve were compared against each other. In conclusion, the mechanism was successfully designed and analyzed.

Reference

[1] Dr. Flavio Firmani, Description of Final Project, 2017. www.sfu.ca
[2] Dr. Flavio Firmani, Finding Roots, 2017. www.sfu.ca
[3] Dr. Flavio Firmani, Optimization, 2017. www.sfu.ca
[4] Dr. Flavio Firmani, PART III—Linear Algebraic Equations, 2017. www.sfu.ca

Appendix

MATLAB code

Newton-Raphson Function

```
function [fx,xr] = NewtRaph(x0,err,a,b,c)
    i = 0;
    xold = x0;
    x = x0;
    while(1)
        i = i + 1;
        fx = lab1fn_Imp(x,a,b,c);
        dfx = funct_deriv(x,a,b);
        x = x - fx/dfx;
        ea = abs((x-xold)/x)*100;
        if ea < err
            xr = x;
            break
        end
        xold = x;
    end
end
```

lab1fn_Imp Function

```
function fx = lab1fn_Imp(x,a,b,c)
    fx = a*cos(x)+b*sin(x)-c;
end
```

funct_deriv Function

```
function dfx = funct_deriv(x,a,b)
    dfx = -a*sin(x)+b*cos(x);
end
```

get_theta3 Function

```
function theta3 = get_theta3(theta4,a,b)
     r4 = 3.70;
    cos_theta3 = (a + r4*cos(theta4));
    sin_theta3 = (b + r4*sin(theta4));
    theta3 = atan2(sin_theta3,cos_theta3);
end
```

Motion Generation and Kinematic Analysis

```
function MSE_211_Lab_2_motion_generation
close all; clear all;
% This is the main file where the inputs and type of problem are
defined

global inputs

%Define the Precision Points (Position and Orientation)
%Position Vectors Px and Py (INPUT POSITIONS)
% project
inputs.Px(1) = 5;
inputs.Py(1) = 4;
inputs.Px(2) = 3.5;
inputs.Py(2) = 4;
inputs.Px(3) = 2;
inputs.Py(3) = 3;
inputs.Px(4) = 1;
inputs.Py(4) = 0;

%Define Orientation in radians (INPUT ANGLES)
%project
inputs.ang(1) = 0; %In radians
inputs.ang(2) = -5*pi/180;   %In radians
inputs.ang(3) = -5*pi/180;   %In radians
inputs.ang(4) = 0;   %In radians

% Desired angles with respect to the first angle (DO NOT CHANGE)
inputs.alpha(1)=inputs.ang(2)-inputs.ang(1);
inputs.alpha(2)=inputs.ang(3)-inputs.ang(1);
inputs.alpha(3)=inputs.ang(4)-inputs.ang(1);

% Desired positions with respect to the first positions (DO NOT CHANGE)
inputs.deltax(1)=inputs.Px(2)-inputs.Px(1);
inputs.deltay(1)=inputs.Py(2)-inputs.Py(1);

inputs.deltax(2)=inputs.Px(3)-inputs.Px(1);
inputs.deltay(2)=inputs.Py(3)-inputs.Py(1);

inputs.deltax(3)=inputs.Px(4)-inputs.Px(1);
inputs.deltay(3)=inputs.Py(4)-inputs.Py(1);

motion_generation %Call motion generation function (see below)
end

function motion_generation

%Optimization Left Side
%Initial values (INPUT INITIAL ESTIMATES)
r2x = 0.58;
r2y = 3.42;
r3ax = 2.5;
r3ay = 2.5;
beta(1)= 10*pi/180;  %In radians
beta(2)= -250*pi/180;  %In radians
beta(3)= -250*pi/180;  %In radians

% Initial esimates in vector form (DO NOT CHANGE)
x0=[r2x,r2y,r3ax,r3ay,beta]';
```

```
%Call Optimization Algorithm (DO NOT CHANGE)
options = optimoptions(@fminunc,'Algorithm','quasi-newton');
[Lx,LFval]= fminunc(@left_side,x0,options); %Call optimization
algorithm

%Optimization Right side
%Initial values (INPUT INITIAL ESTIMATES)
r4x = 0.6;
r4y = 4.4;
r3bx = -2.5;
r3by = 0.5;
gamma(1)= 25*pi/180;  %In radians
gamma(2)= 50*pi/180;  %In radians
gamma(3)= -200*pi/180;  %In radians

% Initial esimates in vector form (DO NOT CHANGE)
x0=[r4x,r4y,r3bx,r3by,gamma]';

%Call Optimization Algorithm (DO NOT CHANGE)
[Rx,RFval]= fminunc(@right_side,x0,options);%Call optimization
algorithm

%Call animation (DO NOT CHANGE)
animation(Lx,Rx,LFval,RFval)
end

function F=left_side(x)
global inputs

%Redefine your variables (DO NOT CHANGE)
r2x=x(1);
r2y=x(2);
r3ax=x(3);
r3ay=x(4);
beta(1)=x(5);
beta(2)=x(6);
beta(3)=x(7);

%WRITE THE RELATIVE EQUATIONS FOR EACH PRECISION POINT
f1x = r2x + r3ax + inputs.deltax(1) - r3ax*cos(inputs.alpha(1)) +
r3ay*sin(inputs.alpha(1)) - r2x*cos(beta(1)) + r2y*sin(beta(1));
f1y = r2y + r3ay + inputs.deltay(1) - r3ay*cos(inputs.alpha(1)) -
r3ax*sin(inputs.alpha(1)) - r2y*cos(beta(1)) - r2x*sin(beta(1));
f2x = r2x + r3ax + inputs.deltax(2) - r3ax*cos(inputs.alpha(2)) +
r3ay*sin(inputs.alpha(2)) - r2x*cos(beta(2)) + r2y*sin(beta(2));
f2y = r2y + r3ay + inputs.deltay(2) - r3ay*cos(inputs.alpha(2)) -
r3ax*sin(inputs.alpha(2)) - r2y*cos(beta(2)) - r2x*sin(beta(2));
f3x = r2x + r3ax + inputs.deltax(3) - r3ax*cos(inputs.alpha(3)) +
r3ay*sin(inputs.alpha(3)) - r2x*cos(beta(3)) + r2y*sin(beta(3));
f3y = r2y + r3ay + inputs.deltay(3) - r3ay*cos(inputs.alpha(3)) -
r3ax*sin(inputs.alpha(3)) - r2y*cos(beta(3)) - r2x*sin(beta(3));


%WRITE YOUR OBJECTIVE FUNCTION
F = sqrt(f1x^2 + f1y^2 + f2x^2 + f2y^2 + f3x^2 + f3y^2);
end

function G=right_side(x)
global inputs
```

```
%Redefine your variables (DO NOT CHANGE)
r4x=x(1);
r4y=x(2);
r3bx=x(3);
r3by=x(4);
gamma(1)=x(5);
gamma(2)=x(6);
gamma(3)=x(7);

%WRITE THE RELATIVE EQUATIONS FOR EACH PRECISION POINT
g1x = r4x + r3bx + inputs.deltax(1) - r3bx*cos(inputs.alpha(1)) +
r3by*sin(inputs.alpha(1)) - r4x*cos(gamma(1)) + r4y*sin(gamma(1));
g1y = r4y + r3by + inputs.deltay(1) - r3by*cos(inputs.alpha(1)) -
r3bx*sin(inputs.alpha(1)) - r4y*cos(gamma(1)) - r4x*sin(gamma(1));
g2x = r4x + r3bx + inputs.deltax(2) - r3bx*cos(inputs.alpha(2)) +
r3by*sin(inputs.alpha(2)) - r4x*cos(gamma(2)) + r4y*sin(gamma(2));
g2y = r4y + r3by + inputs.deltay(2) - r3by*cos(inputs.alpha(2)) -
r3bx*sin(inputs.alpha(2)) - r4y*cos(gamma(2)) - r4x*sin(gamma(2));
g3x = r4x + r3bx + inputs.deltax(3) - r3bx*cos(inputs.alpha(3)) +
r3by*sin(inputs.alpha(3)) - r4x*cos(gamma(3)) + r4y*sin(gamma(3));
g3y = r4y + r3by + inputs.deltay(3) - r3by*cos(inputs.alpha(3)) -
r3bx*sin(inputs.alpha(3)) - r4y*cos(gamma(3)) - r4x*sin(gamma(3));

%WRITE YOUR OBJECTIVE FUNCTION
G = sqrt(g1x^2 + g1y^2 + g2x^2 + g2y^2 + g3x^2 + g3y^2);
end

function animation(Lx,Rx,LFval,RFval)
global inputs
%In this function you might not have to change anything for the lab
%(assuming that you obtain a similar solution to mine). However, you
will
%probably have to make changes for the project, depending on your
outputs.
%THE SECTIONS TO BE CHANGED ARE HIGHLIGHTED WITH LINES / CAPITAL
LETTERS

% Precision Points as a vector
P1=[inputs.Px(1),inputs.Py(1)];
P2=[inputs.Px(2),inputs.Py(2)];
P3=[inputs.Px(3),inputs.Py(3)];
P4=[inputs.Px(4),inputs.Py(4)];

% Set up the screen size
set(0,'Units','pixels');   % Set root units to pixels
dim = get(0,'ScreenSize'); % Get screen size
figure('doublebuffer','on','Position',[0,20,dim(3),dim(4)-100],...
%... line continues in the next row
    'Name','3D Object','NumberTitle','off'); %Create a screen size
figure
set(gcf,'color', [1 1 1]) %Background Colour

%_____________________________________________________________________
%Drawing the box (THIS PART IS ONLY FOR THE LAB)
box_shape = [0,0; 1,0; 1,1; 0,1; 0,0]; %Creating a 2x2 box
box=NaN(5,2,4); %pre-allocating memory
for i=1:4 % rotating and tranlating the box
box(:,:,i)=[cos(inputs.ang(i))*box_shape(:,1)-
sin(inputs.ang(i))*box_shape(:,2),sin(inputs.ang(i))*box_shape(:,1)+cos
```

```
(inputs.ang(i))*box_shape(:,2)] + repmat([inputs.Px(i),
inputs.Py(i)],5,1);
line(box(:,1,i),box(:,2,i)); %ploting the box
end
axis([-4 10 -7 7]); axis equal; hold all;  %CHANGE AXIS SCALE TO FIT
YOUR PROBLEM
%_____________________________________________________________________

% Redefining the variables of the Left side optimization outputs
R2=[Lx(1),Lx(2)]; R3a=[Lx(3),Lx(4)];
beta(1)=Lx(5); beta(2)=Lx(6); beta(3)=Lx(7);

%Finding the length of the links for the left side
r2=norm(R2); r3a=norm(R3a);

% Redefining the variables of the Right Side optimization outputs
R4=[Rx(1),Rx(2)]; R3b=[Rx(3),Rx(4)];
gamma(1)=Rx(5); gamma(2)=Rx(6); gamma(3)=Rx(7);

%Finding the length of the links for the right side
r4=norm(R4); r3b=norm(R3b);

%Location of Ground Pivots
GP2 = P1-R2-R3a;
GP4 = P1-R4-R3b;

% Finding the length of the frame and couple links
r1=norm(GP2-GP4);
r3 = norm((R3a-P1)-(R3b-P1));

% Defining output angles to be between -pi and pi
for i=1:3
if beta(i)>pi; beta(i)=beta(i)-2*pi; end
if beta(i)<-pi; beta(i)=beta(i)+2*pi; end
if gamma(i)>pi; gamma(i)=gamma(i)-2*pi; end
if gamma(i)<pi; gamma(i)=gamma(i)+2*pi; end
end

% Output script of the solution obtained with the optimization
disp(' ')
fprintf('-------------------------------------------------------------
-----------------------------------------\n');
disp(['      LFval           RFval            r1            r2
r3             r4           beta1          beta2          beta3
gamma1       gamma2       gamma3']);
fprintf('-------------------------------------------------------------
-----------------------------------------\n');
ffa=[LFval    RFval   r1 r2 r3 r4 beta(1)*180/pi beta(2)*180/pi
beta(3)*180/pi gamma(1)*180/pi gamma(2)*180/pi gamma(3)*180/pi];
fprintf('%14.8f %14.8f %14.8f %14.8f %14.8f %14.8f %14.8f %14.8f %14.8f
%14.8f %14.8f %14.8f\n',ffa');

% Obtaining input and output angles (theta2 and theta4)
theta2_0= atan2(R2(2),R2(1));
theta2_f=theta2_0+beta(3);
theta4_0= atan2(R4(2),R4(1));
theta4_f=theta4_0+gamma(3);

%Define frame angle (always the same)
theta_1=atan2((GP4(2)-GP2(2)),(GP4(1)-GP2(1)));
```

```
%Ploting
%Plot frame link
line([GP2(1),GP4(1)],[GP2(2),GP4(2)], 'LineWidth', 3, 'Color', 'black')
plot(GP2(1),GP2(2), 'o', 'MarkerFaceColor', [0 0 1], 'MarkerSize',8,
'MarkerEdgeColor','k')
plot(GP4(1),GP4(2), 'o', 'MarkerFaceColor', [0 0 1], 'MarkerSize',8,
'MarkerEdgeColor','k')

%Plot r2 in all four configurations
plot([GP2(1);GP2(1)+r2*cos(theta2_0)],[GP2(2);GP2(2)+r2*sin(theta2_0)],
'b');
plot([GP2(1);GP2(1)+r2*cos(theta2_0+beta(1))],[GP2(2);GP2(2)+r2*sin(the
ta2_0+beta(1))],'b');
plot([GP2(1);GP2(1)+r2*cos(theta2_0+beta(2))],[GP2(2);GP2(2)+r2*sin(the
ta2_0+beta(2))],'b');
plot([GP2(1);GP2(1)+r2*cos(theta2_0+beta(3))],[GP2(2);GP2(2)+r2*sin(the
ta2_0+beta(3))],'b');

%Plot r4 in all four configurations
plot([GP4(1);GP4(1)+r4*cos(theta4_0)],[GP4(2);GP4(2)+r4*sin(theta4_0)],
'r');
plot([GP4(1);GP4(1)+r4*cos(theta4_0+gamma(1))],[GP4(2);GP4(2)+r4*sin(th
eta4_0+gamma(1))],'r');
plot([GP4(1);GP4(1)+r4*cos(theta4_0+gamma(2))],[GP4(2);GP4(2)+r4*sin(th
eta4_0+gamma(2))],'r');
plot([GP4(1);GP4(1)+r4*cos(theta4_0+gamma(3))],[GP4(2);GP4(2)+r4*sin(th
eta4_0+gamma(3))],'r');

%Plot r3 in all four configurations
plot([GP2(1)+r2*cos(theta2_0);GP4(1)+r4*cos(theta4_0);P1(1);GP2(1)+r2*c
os(theta2_0)],[GP2(2)+r2*sin(theta2_0);GP4(2)+r4*sin(theta4_0);P1(2);GP
2(2)+r2*sin(theta2_0)],'g')
plot([GP2(1)+r2*cos(theta2_0+beta(1));GP4(1)+r4*cos(theta4_0+gamma(1));
P2(1);GP2(1)+r2*cos(theta2_0+beta(1))],[GP2(2)+r2*sin(theta2_0+beta(1))
;GP4(2)+r4*sin(theta4_0+gamma(1));P2(2);GP2(2)+r2*sin(theta2_0+beta(1))
],'g')
plot([GP2(1)+r2*cos(theta2_0+beta(2));GP4(1)+r4*cos(theta4_0+gamma(2));
P3(1);GP2(1)+r2*cos(theta2_0+beta(2))],[GP2(2)+r2*sin(theta2_0+beta(2))
;GP4(2)+r4*sin(theta4_0+gamma(2));P3(2);GP2(2)+r2*sin(theta2_0+beta(2))
],'g')
plot([GP2(1)+r2*cos(theta2_0+beta(3));GP4(1)+r4*cos(theta4_0+gamma(3));
P4(1);GP2(1)+r2*cos(theta2_0+beta(3))],[GP2(2)+r2*sin(theta2_0+beta(3))
;GP4(2)+r4*sin(theta4_0+gamma(3));P4(2);GP2(2)+r2*sin(theta2_0+beta(3))
],'g')

%_____
____
%Definine range of theta2 (MAKE SURE IT WILL ROTATE IN THE RIGHT
DIRECTION)
%The mechanism can rotate clockwise or counterclockwise, also the
magnitude
%of the angle is important for example if you move counterclockwise
from
%theta2_0=100 (deg) to theta2_f= 20 (deg), then you have to add 360 deg
to
%theta2_f, as shown below
theta_2=theta2_0:(2*pi+theta2_f-theta2_0)/100:2*pi+theta2_f;
%_____
____
```

```
%_____________________________________________________________________
____
%If the coupler plate is inverted in your animation, ADD A NEGATIVE
SIGN TO
%THE phi angle, i.e., phi=-acos(...)
%Defining coupler plate
R3=R3a-R3b;
phi=-acos(dot([R3a(1) R3a(2)],[R3(1) R3(2)])/(r3*r3a));
r3a=norm(R3a);
%_____________________________________________________________________
____

% Pre-allocating memory
Theta_3=NaN(length(theta_2),1); Theta_4=NaN(length(theta_2),1);

%Drawing the staircase
steps = [0,0; 5,0; 5,4; 8,4; 8,4];
line(steps(:,1),steps(:,2), 'LineWidth', 2, 'color', 'black'); axis
equal
axis([-4 10 -7 7]); axis equal; hold all;  %CHANGE AXIS SCALE TO FIT
YOUR PROBLEM


grid minor
%Animating Mechanism
for i=1:length(theta_2)
    %Displacement Analysis
    a = r1*cos(theta_1) - r2*cos(theta_2(i));
    b = r1*sin(theta_1) - r2*sin(theta_2(i));
    c = (r3^2 - a^2 -r4^2 - b^2)/(2*r4);
    [fx,theta_4] = NewtRaph(atan2(b,a) + atan2(sqrt(a^2+b^2-
c^2),c),10E-6,a,b,c);
    theta_3 = get_theta3(theta_4,a,b);

    %Plotting Links

R(2)=line([GP2(1),GP2(1)+r2*cos(theta_2(i))],[GP2(2),GP2(2)+r2*sin(thet
a_2(i))], 'LineWidth', 3);
    R(3)=line([GP2(1)+r2*cos(theta_2(i)),
GP2(1)+r2*cos(theta_2(i))+r3a*cos(theta_3-phi),
GP4(1)+r4*cos(theta_4),GP2(1)+r2*cos(theta_2(i))],[GP2(2)+r2*sin(theta_
2(i)),GP2(2)+r2*sin(theta_2(i))+r3a*sin(theta_3-phi),
GP4(2)+r4*sin(theta_4),GP2(2)+r2*sin(theta_2(i))], 'LineWidth', 2);
    R(4)=line([GP4(1),
GP4(1)+r4*cos(theta_4)],[GP4(2),GP4(2)+r4*sin(theta_4)], 'LineWidth',
3);

    %Filling translucent Plate
    Plate=fill([GP2(1)+r2*cos(theta_2(i)),
GP2(1)+r2*cos(theta_2(i))+r3a*cos(theta_3-phi),
GP4(1)+r4*cos(theta_4),GP2(1)+r2*cos(theta_2(i))],[GP2(2)+r2*sin(theta_
2(i)),GP2(2)+r2*sin(theta_2(i))+r3a*sin(theta_3-phi),
GP4(2)+r4*sin(theta_4),GP2(2)+r2*sin(theta_2(i))],'b');
    set(Plate,'facealpha',0.5)

    %Plotting joints
    C(1)=plot(GP2(1)+r2*cos(theta_2(i)),GP2(2)+r2*sin(theta_2(i)), 'o',
'MarkerFaceColor', [0 0 1], 'MarkerSize',8, 'MarkerEdgeColor','k');
    C(2)=plot(GP4(1)+r4*cos(theta_4),GP4(2)+r4*sin(theta_4), 'o',
'MarkerFaceColor', [0 0 1], 'MarkerSize',8, 'MarkerEdgeColor','k');
```

```
    %_____________________________________________
    % Scaling and pausing animation SCALE YOUR ANIMATION BASED ON YOUR
WORKSPACE
    axis equal; axis([-4 10 -7 7])
    pause(0.1)
    %_____________________________________________

    % Turn off links and joints
    if i < length(theta_2)
        for ii=2:4; set(R(ii),'visible','off'); end
        set(C,'visible','off')
        set(Plate,'visible','off')
    end

    % Collect all Thetas (In case you want to plot them)
    Theta_3(i)=theta_3;
    Theta_4(i)=theta_4;
end

close all

theta2 = theta_2;
theta_3 = Theta_3;
theta_4 = Theta_4;

figure (1)
title('Angular Displacement of link R3 and R4');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Angular Displacement (deg)'); % y-axis label
hold on;
plot(radtodeg(theta2),radtodeg(theta_3),radtodeg(theta2),radtodeg(theta
_4))
legend('Angular Displacement (R3)','Angular Displacement (R4)')

%Find center of mass
for i = 1:length(theta2)
    x2(i) = (r2/2)*cos(theta2(i));
    y2(i) = (r2/2)*sin(theta2(i));
    x3(i) = r2*cos(theta2(i))+(r3/2)*cos(theta_3(i));
    y3(i) = r2*sin(theta2(i))+(r3/2)*sin(theta_3(i));
    x4(i) = (r4/2)*cos(theta_4(i));
    y4(i) = (r4/2)*sin(theta_4(i));
end

figure (2)
title('Center of Mass of Link R2');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Position'); % y-axis label
hold on;
plot(radtodeg(theta2),x2,radtodeg(theta2),y2)
legend('x2','y2')

figure (3)
title('Center of Mass of Link R3');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Position'); % y-axis label
hold on;
plot(radtodeg(theta2),x3,radtodeg(theta2),y3)
legend('x3','y3')
```

```
figure (4)
title('Center of Mass of Link R4');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Position'); % y-axis label
hold on;
plot(radtodeg(theta2),x4,radtodeg(theta2),y4)
legend('x4','y4')

%Analytic Solution
%Angular Velocity and Angular Acceleration
theta2_v = 1;    %theta2. = 1 rad/s

for i = 1:length(theta2)
   A = [-r3*sin(theta_3(i)) r4*sin(theta_4(i));
       r3*cos(theta_3(i)) -r4*cos(theta_4(i))];
   B = [r2*sin(theta2(i))*theta2_v;
       -r2*cos(theta2(i))*theta2_v];
   C = A\B;
   theta3_v(i) = C(1);
   theta4_v(i) = C(2);
   D =
[r2*cos(theta2(i))*(theta2_v)^2+r3*cos(theta_3(i))*(theta3_v(i))^2-
r4*cos(theta_4(i))*(theta4_v(i))^2;

r2*sin(theta2(i))*(theta2_v)^2+r3*sin(theta_3(i))*(theta3_v(i))^2-
r4*sin(theta_4(i))*(theta4_v(i))^2];
   E = A\D;
   theta3_a(i) = E(1);
   theta4_a(i) = E(2);
end

figure (5)
title('Angular Velocty of link R3 and R4 (Analytic Solution)');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Angular Velocity (deg/s)'); % y-axis label
hold on;
plot(radtodeg(theta2),radtodeg(theta3_v),radtodeg(theta2),radtodeg(thet
a4_v))
legend('Angular Velocity (R3)','Angular Velocity (R4)')

figure (6)
title('Angular Acceleration of link R3 and R4 (Analytic Solution)');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Angular Acceleration (deg/s)'); % y-axis label
hold on;
plot(radtodeg(theta2),radtodeg(theta3_a),radtodeg(theta2),radtodeg(thet
a4_a))
legend('Angular Acceleration (R3)','Angular Acceleration (R4)')

%Linear Velocity and Linear Acceleration
for i = 1:length(theta2)
    v2x(i) = -(r2/2)*theta2_v*sin(theta2(i));
    v2y(i) = (r2/2)*theta2_v*cos(theta2(i));
    v3x(i) = -r2*theta2_v*sin(theta2(i))-
(r3/2)*theta3_v(i)*sin(theta_3(i));
    v3y(i) =
r2*theta2_v*cos(theta2(i))+(r3/2)*theta3_v(i)*cos(theta_3(i));
    v4x(i) = -(r4/2)*theta4_v(i)*sin(theta_4(i));
    v4y(i) = (r4/2)*theta4_v(i)*cos(theta_4(i));
    a2x(i) = -(r2/2)*cos(theta2(i))*(theta2_v)^2;
```

```
    a2y(i) = -(r2/2)*sin(theta2(i))*(theta2_v)^2;
    a3x(i) = -r2*cos(theta2(i))*(theta2_v)^2-
(r3/2)*sin(theta_3(i))*theta3_a(i)-
(r3/2)*cos(theta_3(i))*(theta3_v(i))^2;
    a3y(i) = -
r2*sin(theta2(i))*(theta2_v)^2+(r3/2)*cos(theta_3(i))*theta3_a(i)-
(r3/2)*sin(theta_3(i))*(theta3_v(i))^2;
    a4x(i) = -(r4/2)*sin(theta_4(i))*theta4_a(i)-
(r4/2)*cos(theta_4(i))*(theta4_v(i))^2;
    a4y(i) = (r4/2)*cos(theta_4(i))*theta4_a(i)-
(r4/2)*sin(theta_4(i))*(theta4_v(i))^2;
end

figure (7)
title('Linear Velocity of the center of mass of link R2 (Analytic
Solution)');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Linear Velocity (m/s)'); % y-axis label
hold on;
plot(radtodeg(theta2),v2x,radtodeg(theta2),v2y)
legend('v2x','v2y')

figure (8)
title('Linear Velocity of the center of mass of link R3 (Analytic
Solution)');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Linear Velocity (m/s)'); % y-axis label
hold on;
plot(radtodeg(theta2),v3x,radtodeg(theta2),v3y)
legend('v3x','v3y')

figure (9)
title('Linear Velocity of the center of mass of link R4 (Analytic
Solution)');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Linear Velocity (m/s)'); % y-axis label
hold on;
plot(radtodeg(theta2),v4x,radtodeg(theta2),v4y)
legend('v4x','v4y')

figure (10)
title('Linear Acceleration of the center of mass of link R2 (Analytic
Solution)');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Linear Acceleration (m/s)'); % y-axis label
hold on;
plot(radtodeg(theta2),a2x,radtodeg(theta2),a2y)
legend('a2x','a2y')

figure (11)
title('Linear Acceleration of the center of mass of link R3 (Analytic
Solution)');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Linear Acceleration (m/s)'); % y-axis label
hold on;
plot(radtodeg(theta2),a3x,radtodeg(theta2),a3y)
legend('a3x','a3y')

figure (12)
title('Linear Acceleration of the center of mass of link R4 (Analytic
Solution)');
```

```
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Linear Acceleration (m/s)'); % y-axis label
hold on;
plot(radtodeg(theta2),a4x,radtodeg(theta2),a4y)
legend('a4x','a4y')

%Numerical Solution
%Center Finite-Difference (Basic Version)
h = pi/36;
for i = 2:length(theta2)-1
    theta2_num(i-1) = theta2(i);    %Shifting theta2 element
    %Angular Velocity
    theta3_v_num(i-1) = (theta_3(i+1)-theta_3(i-1))/(2*h);
    theta4_v_num(i-1) = (theta_4(i+1)-theta_4(i-1))/(2*h);
    %Angular Aceleration
    theta3_a_num(i-1) = (theta_3(i+1)-2*theta_3(i)+theta_3(i-1))/(h^2);
    theta4_a_num(i-1) = (theta_4(i+1)-2*theta_4(i)+theta_4(i-1))/(h^2);
    %Linear Velocity
    v2x_num(i-1) = (x2(i+1)-x2(i-1))/(2*h);
    v2y_num(i-1) = (y2(i+1)-y2(i-1))/(2*h);
    v3x_num(i-1) = (x3(i+1)-x3(i-1))/(2*h);
    v3y_num(i-1) = (y3(i+1)-y3(i-1))/(2*h);
    v4x_num(i-1) = (x4(i+1)-x4(i-1))/(2*h);
    v4y_num(i-1) = (y4(i+1)-y4(i-1))/(2*h);
    %Linear Acceleration
    a2x_num(i-1) = (x2(i+1)-2*x2(i)+x2(i-1))/(h^2);
    a2y_num(i-1) = (y2(i+1)-2*y2(i)+y2(i-1))/(h^2);
    a3x_num(i-1) = (x3(i+1)-2*x3(i)+x3(i-1))/(h^2);
    a3y_num(i-1) = (y3(i+1)-2*y3(i)+y3(i-1))/(h^2);
    a4x_num(i-1) = (x4(i+1)-2*x4(i)+x4(i-1))/(h^2);
    a4y_num(i-1) = (y4(i+1)-2*y4(i)+y4(i-1))/(h^2);
end

figure (13)
title('Angular Velocty of link R3 and R4 (Numerical Solution: Basic
Version)');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Angular Velocity (deg/s)'); % y-axis label
hold on;
plot(radtodeg(theta2_num),radtodeg(theta3_v_num),radtodeg(theta2_num),r
adtodeg(theta4_v_num))
legend('Angular Velocity (R3)','Angular Velocity (R4)')

figure (14)
title('Angular Acceleration of link R3 and R4 (Numerical Solution:
Basic Version)');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Angular Acceleration (deg/s)'); % y-axis label
hold on;
plot(radtodeg(theta2_num),radtodeg(theta3_a_num),radtodeg(theta2_num),r
adtodeg(theta4_a_num))
legend('Angular Acceleration (R3)','Angular Acceleration (R4)')

figure (15)
title('Linear Velocity of the center of mass of link R2 (Numerical
Solution: Basic Version)');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Linear Velocity (m/s)'); % y-axis label
hold on;
plot(radtodeg(theta2_num),v2x_num,radtodeg(theta2_num),v2y_num)
legend('v2x','v2y')
```

```
figure (16)
title('Linear Velocity of the center of mass of link R3 (Numerical
Solution: Basic Version)');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Linear Velocity (m/s)'); % y-axis label
hold on;
plot(radtodeg(theta2_num),v3x_num,radtodeg(theta2_num),v3y_num)
legend('v3x','v3y')

figure (17)
title('Linear Velocity of the center of mass of link R4 (Numerical
Solution: Basic Version)');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Linear Velocity (m/s)'); % y-axis label
hold on;
plot(radtodeg(theta2_num),v4x_num,radtodeg(theta2_num),v4y_num)
legend('v4x','v4y')

figure (18)
title('Linear Acceleration of the center of mass of link R2 (Numerical
Solution: Basic Version)');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Linear Acceleration (m/s)'); % y-axis label
hold on;
plot(radtodeg(theta2_num),a2x_num,radtodeg(theta2_num),a2y_num)
legend('a2x','a2y')

figure (19)
title('Linear Acceleration of the center of mass of link R3 (Numerical
Solution: Basic Version)');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Linear Acceleration (m/s)'); % y-axis label
hold on;
plot(radtodeg(theta2_num),a3x_num,radtodeg(theta2_num),a3y_num)
legend('a3x','a3y')

figure (20)
title('Linear Acceleration of the center of mass of link R4 (Numerical
Solution: Basic Version)');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Linear Acceleration (m/s)'); % y-axis label
hold on;
plot(radtodeg(theta2_num),a4x_num,radtodeg(theta2_num),a4y_num)
legend('a4x','a4y')

%Center Finite-Difference (Expanded Version)
%h = pi/36;
for i = 3:length(theta2)-2
    theta2_numE(i-2) = theta2(i);    %Shifting theta2 element
    %Angular Velocity
    theta3_v_numE(i-2) = (-theta_3(i+2)+8*theta_3(i+1)-8*theta_3(i-
1)+theta_3(i-2))/(12*h);
    theta4_v_numE(i-2) = (-theta_4(i+2)+8*theta_4(i+1)-8*theta_4(i-
1)+theta_4(i-2))/(12*h);
    %Angular Aceleration
    theta3_a_numE(i-2) = (-theta_3(i+2)+16*theta_3(i+1)-
30*theta_3(i)+16*theta_3(i-1)-theta_3(i-2))/(12*h^2);
    theta4_a_numE(i-2) = (-theta_4(i+2)+16*theta_4(i+1)-
30*theta_4(i)+16*theta_4(i-1)-theta_4(i-2))/(12*h^2);
    %Linear Velocity
```

```
    v2x_numE(i-2) = (-x2(i+2)+8*x2(i+1)-8*x2(i-1)+x2(i-2))/(12*h);
    v2y_numE(i-2) = (-y2(i+2)+8*y2(i+1)-8*y2(i-1)+y2(i-2))/(12*h);
    v3x_numE(i-2) = (-x3(i+2)+8*x3(i+1)-8*x3(i-1)+x3(i-2))/(12*h);
    v3y_numE(i-2) = (-y3(i+2)+8*y3(i+1)-8*y3(i-1)+y3(i-2))/(12*h);
    v4x_numE(i-2) = (-x4(i+2)+8*x4(i+1)-8*x4(i-1)+x4(i-2))/(12*h);
    v4y_numE(i-2) = (-y4(i+2)+8*y4(i+1)-8*y4(i-1)+y4(i-2))/(12*h);
    %Linear Acceleration
    a2x_numE(i-2) = (-x2(i+2)+16*x2(i+1)-30*x2(i)+16*x2(i-1)-x2(i-
2))/(12*h^2);
    a2y_numE(i-2) = (-y2(i+2)+16*y2(i+1)-30*y2(i)+16*y2(i-1)-y2(i-
2))/(12*h^2);
    a3x_numE(i-2) = (-x3(i+2)+16*x3(i+1)-30*x3(i)+16*x3(i-1)-x3(i-
2))/(12*h^2);
    a3y_numE(i-2) = (-y3(i+2)+16*y3(i+1)-30*y3(i)+16*y3(i-1)-y3(i-
2))/(12*h^2);
    a4x_numE(i-2) = (-x4(i+2)+16*x4(i+1)-30*x4(i)+16*x4(i-1)-x4(i-
2))/(12*h^2);
    a4y_numE(i-2) = (-y4(i+2)+16*y4(i+1)-30*y4(i)+16*y4(i-1)-y4(i-
2))/(12*h^2);
end

figure (21)
title('Angular Velocty of link R3 and R4 (Numerical Solution: Expanded
Version)');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Angular Velocity (deg/s)'); % y-axis label
hold on;
plot(radtodeg(theta2_numE),radtodeg(theta3_v_numE),radtodeg(theta2_numE
),radtodeg(theta4_v_numE))
legend('Angular Velocity (R3)','Angular Velocity (R4)')

figure (22)
title('Angular Acceleration of link R3 and R4 (Numerical Solution:
Expanded Version)');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Angular Acceleration (deg/s)'); % y-axis label
hold on;
plot(radtodeg(theta2_numE),radtodeg(theta3_a_numE),radtodeg(theta2_numE
),radtodeg(theta4_a_numE))
legend('Angular Acceleration (R3)','Angular Acceleration (R4)')

figure (23)
title('Linear Velocity of the center of mass of link R2 (Numerical
Solution: Expanded Version)');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Linear Velocity (m/s)'); % y-axis label
hold on;
plot(radtodeg(theta2_numE),v2x_numE,radtodeg(theta2_numE),v2y_numE)
legend('v2x','v2y')

figure (24)
title('Linear Velocity of the center of mass of link R3 (Numerical
Solution: Expanded Version)');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Linear Velocity (m/s)'); % y-axis label
hold on;
plot(radtodeg(theta2_numE),v3x_numE,radtodeg(theta2_numE),v3y_numE)
legend('v3x','v3y')

figure (25)
title('Linear Velocity of the center of mass of link R4 (Numerical
```

```
Solution: Expanded Version)');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Linear Velocity (m/s)'); % y-axis label
hold on;
plot(radtodeg(theta2_numE),v4x_numE,radtodeg(theta2_numE),v4y_numE)
legend('v4x','v4y')

figure (26)
title('Linear Acceleration of the center of mass of link R2 (Numerical
Solution: Expanded Version)');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Linear Acceleration (m/s)'); % y-axis label
hold on;
plot(radtodeg(theta2_numE),a2x_numE,radtodeg(theta2_numE),a2y_numE)
legend('a2x','a2y')

figure (27)
title('Linear Acceleration of the center of mass of link R3 (Numerical
Solution: Expanded Version)');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Linear Acceleration (m/s)'); % y-axis label
hold on;
plot(radtodeg(theta2_numE),a3x_numE,radtodeg(theta2_numE),a3y_numE)
legend('a3x','a3y')

figure (28)
title('Linear Acceleration of the center of mass of link R4 (Numerical
Solution: Expanded Version)');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Linear Acceleration (m/s)'); % y-axis label
hold on;
plot(radtodeg(theta2_numE),a4x_numE,radtodeg(theta2_numE),a4y_numE)
legend('a4x','a4y')

% Dynamics Analysis
m1=0.456; m2=0.3764; m3=0.4762; m4=0.4308;
i1=0.5836; i2=0.3282; i3=0.6646; i4=0.4920;

r21x=x2 ; r21y=y2;
r32x=x3 ; r32y=y3;
r23x=-r32x ; r23y=-r32y;
r43x=x4 ; r43y=y4;
r34x=-r43x ; r34y=-r43y;
r41x=r4-x4 ; r41y=r4-y4;

ma2x=m2*a2x; ma2y=m2*a2y;
ma3x=m3*a3x; ma3y=m3*a3y;
ma4x=m4*a4x; ma4y=m4*a4y;
it3=i3*theta3_a; it4=i4*theta4_a;

for i = 1:length(theta2)
D = [ 1 0 1 0 0 0 0 0 0;
      0 1 0 1 0 0 0 0 0;
      r21y(i) -r21x(i) -r23y(i) r23x(i) 0 0 0 0 1;
      0 0 -1 0 1 0 0 0 0;
      0 0 0 -1 0 1 0 0 0;
      0 0 -r32y(i) -r32x(i) -r34y(i) -r34x(i) 0 0 0;
      0 0 0 0 -1 0 1 0 0;
      0 0 0 0 0 -1 0 1 0;
      0 0 0 0 r43y(i) -r43x(i) r41y(i) -r41x(i) 0];
```

```
Y = [ma2x(i);ma2y(i);0;ma3x(i);ma3y(i);it3(i);ma4x(i);ma4y(i);it4(i)];

namics = inv(D)*Y;
M12(i)=namics(end);
end

figure (29)
plot(radtodeg(theta2),M12);
title('Torque change with theta 2');
xlabel('Theta2 (deg)'); % x-axis label
ylabel('Torque'); % y-axis label


close all;

% coupler curve
theta2_ext=theta2(1:3:end/3);

x2_ext=x2(1:3:end/3);
y2_ext=y2(1:3:end/3);

x3_ext=x3(1:3:end/3);
y3_ext=y3(1:3:end/3);

x4_ext=x4(1:3:end/3);
y4_ext=y4(1:3:end/3);

for i = 2:4
figure (29+i)
hold on;
    if i==2
        x=x2_ext;
        y=y2_ext;
        %linear regression
        y1=0.9839*x+1.578;
        plot(x,y1,'--');
    end
    if i==3
        x=x3_ext;
        y=y3_ext;
        y1=1.031*x+3.9969;
        plot(x,y1,'--');
    end
    if i==4
        x=x4_ext;
        y=y4_ext;
        y1=1.0577*x+2.1331;
        plot(x,y1,'--');
    end

plot(x,y);  % exact points

%polynomial regression
p2 = polyfit(x,y,2);
y2=p2(3)+p2(2)*x+p2(1)*x.^2;
plot(x,y2,'--');

p3 = polyfit(x,y,3);
```

```
y3=p3(4)+p3(3)*x+p3(2)*x.^2+p3(1)*x.^3;
plot(x,y3,'--');

p4 = polyfit(x,y,4);
y4=p4(5)+p4(4)*x+p4(3)*x.^2+p4(2)*x.^3+p4(1)*x.^4;
plot(x,y4,'--');
legend('linear','actual','degree: 2','degree: 3','degree: 4');
    if i==2
        title('Coupler Curve of link 2');
    end
    if i==3
        title('Coupler Curve of link 3');
    end
    if i==4
        title('Coupler Curve of link 4');
    end
end


% Interpolation
for n = 2:4
    if n==2
        x=x2_ext;
        y=y2_ext;
    end
    if n==3
        x=x3_ext;
        y=y3_ext;
    end
    if n==4
        x=x4_ext;
        y=y4_ext;
    end
% First Divided Differences
for i = 2:length(x)
    fdd(i-1)=(y(i)-y(i-1))/(x(i)-x(i-1));
end
% Second Divided Differences
for i = 2:length(fdd)
    sdd(i-1)=(fdd(i)-fdd(i-1))/(x(i+1)-x(i-1));
end

% Third Divided Differences
for i = 2:length(sdd)
    tdd(i-1)=(sdd(i)-sdd(i-1))/(x(i+2)-x(i-1));
end

% Fourth Divided Differences
for i = 2:length(tdd)
    fodd(i-1)=(tdd(i)-tdd(i-1))/(x(i+3)-x(i-1));
end

% Polynomial
% Second degree
% for i = 2:length(sdd)
%     xi(i-1)=(x(i)+x(i-1))/2;
%     yi(i-1)=y(i-1)+fdd(i-1)*(xi(i-1)-x(i-1))+sdd(i-1)*(xi(i-1)-x(i-
1))*(xi(i-1)-x(i));
% end
```

```
% % third degree
% for i = 2:length(tdd)
%     xi(i-1)=(x(i)+x(i-1))/2;
%     yi(i-1)=y(i-1)+fdd(i-1)*(xi(i-1)-x(i-1))+sdd(i-1)*(xi(i-1)-x(i-
1))*(xi(i-1)-x(i))+tdd(i-1)*(xi(i-1)-x(i-1))*(xi(i-1)-x(i))*(xi(i-1)-
x(i+1));
% end

% % Fourth degree
for i = 2:length(fodd)
    xi(i-1)=(x(i)+x(i-1))/2;
    yi(i-1)=y(i-1)+fdd(i-1)*(xi(i-1)-x(i-1))+sdd(i-1)*(xi(i-1)-x(i-
1))*(xi(i-1)-x(i))+tdd(i-1)*(xi(i-1)-x(i-1))*(xi(i-1)-x(i))*(xi(i-1)-
x(i+1))+fodd(i-1)*(xi(i-1)-x(i-1))*(xi(i-1)-x(i))*(xi(i-1)-
x(i+1))*(xi(i-1)-x(i+2));
end

figure(32+n);
plot(x,y,'x-',xi,yi,'o');
legend('Extracted Point on Coupler Curve','Interpolation');
    if n==2
        title('Interpolation of link 2');
    end
    if n==3
        title('Interpolation of link 3');
    end
    if n==4
        title('Interpolation of link 4');
    end
end
end
```