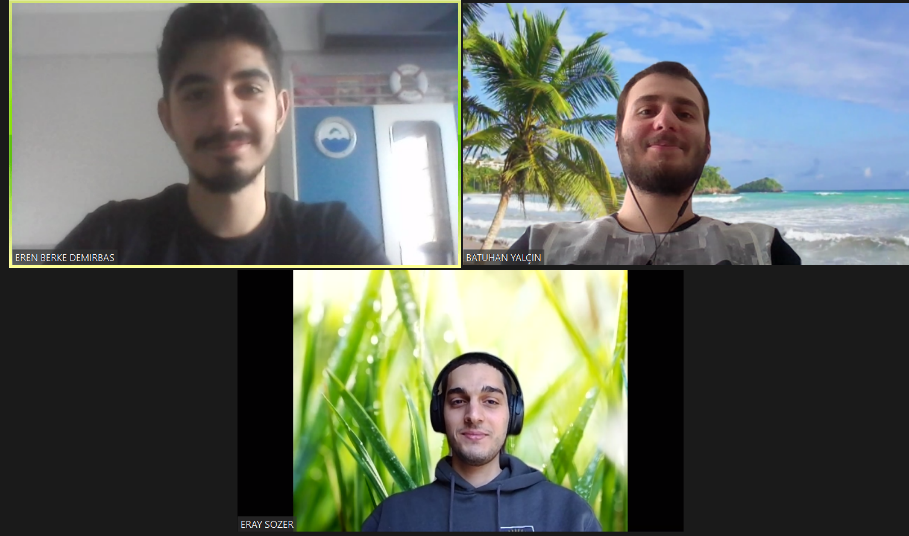
GROUP #17: Fowler Flap

MECH 206 Dynamics Project Report



Batuhan Yalçın

Eray Sözer

Eren Berke Demirbaş

Responsible TA: Denizhan Korkut

Date 25.05.2020



**Contents**

**1. Introduction………………………………………………………………………………..**

**2. Technical Background………………………………………………………………….**

**3. CAD Design………………………………………………………………………………….**

**4. Mechanism Actuation and Manufacturing…………………………………..**

**5. Motion Simulation……………………………………………………………………….**

**6. Derivation of Dynamic Equations…………………………………………………**

**6.1. Planar Kinematics…………………………………………………………….**

**6.2. Planar Kinetics: Force and Acceleration…………………………….**

**6.3. Planar Kinetics: Work and Energy……………………………………..**

**7. MATLAB Analysis………………………………………………………………………….**

**8. Experimental Validation……………………………………………………………….**

**9. Conclusion…………………………………………………………………………………….**

**10. References………………………………………………………………………………….**

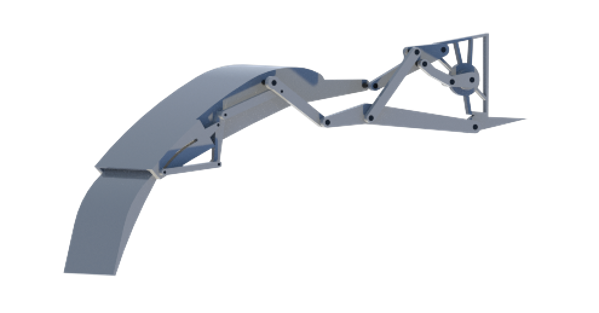
**11. Appendix……………………………………………………………………………………**

**1. Introduction**

-Our system is the fowler flaps

-İt is mainly using when during the take-off and landing operations of the planes

-Main idea of the operation principle is the increasing the area of the wings, with that strategy increasing the lifting forces and drag forces. These process helps an airplane to easily take-off and gives the more shorter requirement distances to landing process

f your mechanism



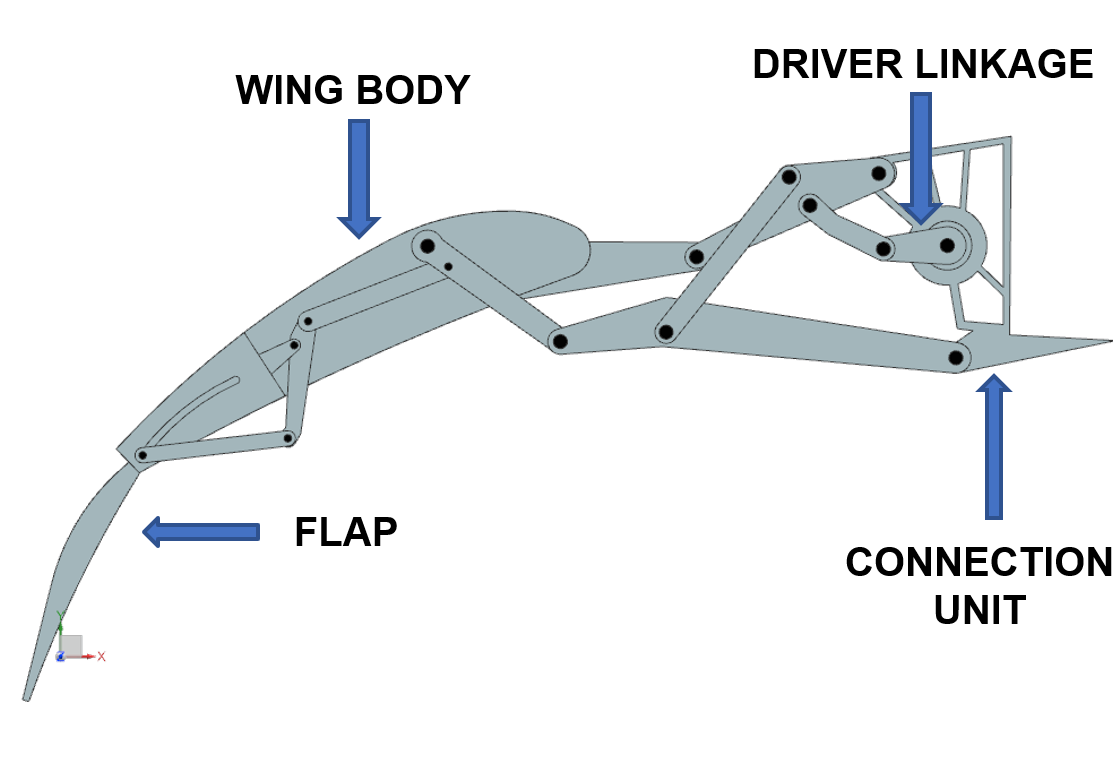
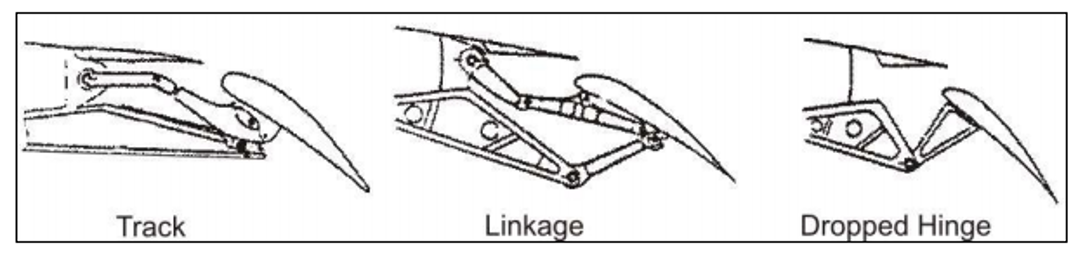
**2. Technical Background**

-Our system has 9 bar linkage and 15 joints between the bars and we have 1 slider mechanism also in the our system.

-Our system has mainly 4 part: Connection unit, driver linkage, wing body, and the flap

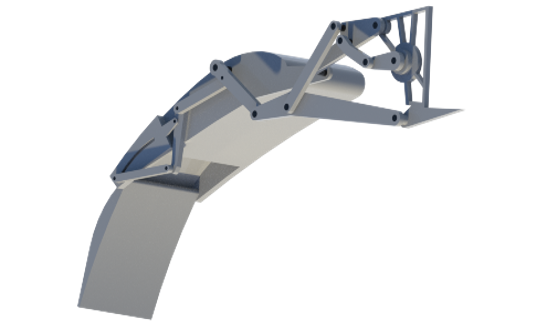
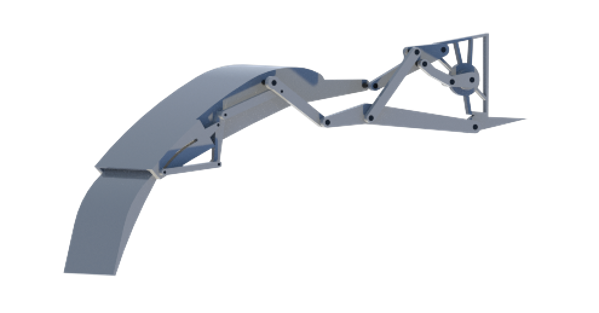
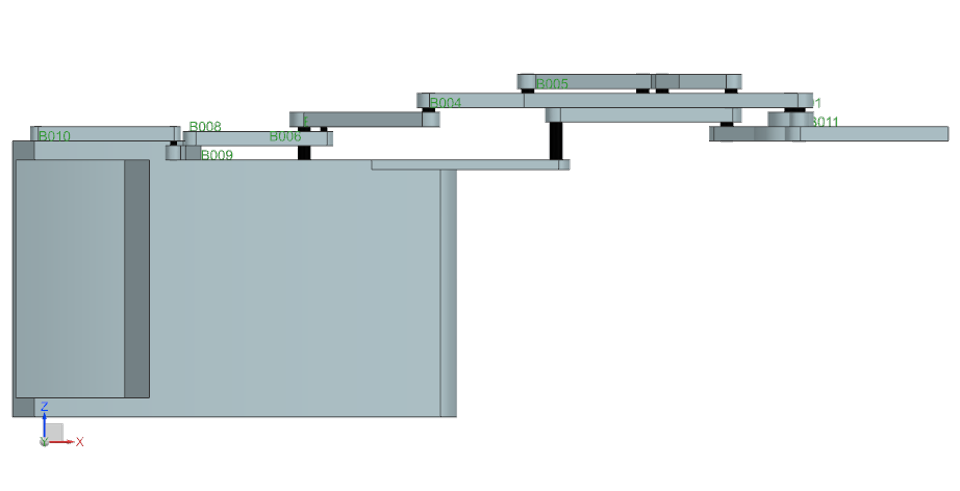
-Our system has the 1 degree of freedom since that using one motor were enough to our system.

-As you see below there were an types of the system for the fowler flap but lots of them include the pistons to the best suitable case for the our courses and the manufacturing easily we choose the fully linkage system. Cite in the picture

[](https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.fzt.haw-hamburg.de%2Fpers%2FScholz%2FHOOU%2FAircraftDesign_8_HighLift.pdf&psig=AOvVaw1a787dUDw3Z1epnnupbafy&ust=1590496560057000&source=images&cd=vfe&ved=0CAIQjRxqFwoTCIDO68iDz-kCFQAAAAAdAAAAABAD)

**3. CAD Design**

-In the market there were lots of different types systems. Our design completely has a linkage system, in other systems there were also a piston which mainly using for the decreasing the force on the motor. Since we do not use the piston, we ignore the thermodynamics effects on the pistons and with the linkage system we find more precise output on the system.



**4. Mechanism Actuation and Manufacturing**

-If we were doing this project in school, mainly we were using the laser cut technique for the cutting the Plexiglass to get necessary shape

. We also choose the 0.6 mm plexiglass to get more strength links with higher wall thickness.

. In real life according our research references given in the reference page to the system especially for the flap part their using the same material with wing itself and it is mostly aluminum or composites such as carbon fiber, Kevlar fiber…

. In real life process is the first Forging manufacturing to giving first shape of the aluminum then the machining process with the CNC and other operators to obtain more smother surfaces which is more suitable for the air conditions.

-If we were doing this project in real life for 3D mechanism, we were using the differansiyel. Getting same torque on the both side of the differencial could be the our major challenge on the system.

-We get the necessary torque for the system from the Nx which is were 47.6 N.mm

1 Kg.cm torque= 98.1 N.mm

\*\*-From the market we found the 0.5Kg.cm torque motors in the common using which were close our necessary torque. We choice brushless DC motor for our system, to getting high tolerance, performances and the cheaper one in real life applications servo motors are using but with the brushless DC motor we are choosing life to money and todays computersgtechnology we can reach the necessary tolerance in the control of brushless DC Motor.

-We are now the density of the material and the total volume of the material from the Nx and dencity\*volume= mass and with that one we found total weight is to 1.2 KG.

-We watched almost all the videos of Boeing’s plane and the operation was taking in the 13 second so we design our system to everything happen in 13 second.

Bills of Materials:

|  |  |
| --- | --- |
| Product | Price (TL) |
| 6mm Plexiglass 30x30 cm^2 | [54.17](https://urun.n11.com/diger/seffaf-pleksi-pleksiglas-levha-58-6mm-seffaf-pleksi-P398679953?Ebat+Se%C3%A7imi=40CM+X+40CM&gclid=CjwKCAjwtqj2BRBYEiwAqfzur-dWhXmpyD46ii681F5-v3wvHSQKXwLqm38M7WPAcEEkulZw_LHyjRoCLO4QAvD_BwE&gclsrc=aw.ds) |
| 30mm Wooden 100x30 cm^2 | [120](https://www.hepsiburada.com/kilizman-masif-ahsap-plaka-30-cm-en-80-cm-boy-30-mm-levha-kalinligi-pm-HB0000013TQN) |
| Brushless DC Motor | [102.71](https://www.direnc.net/cf2822-1200kvdc-fircasiz-motor-emax?language=tr&h=93d10ef6&gclid=CjwKCAjwtqj2BRBYEiwAqfzur_P1KR0S4wI07tglbfQql7FYyHHybaDOSGYsmHo8xe8CKox6P1cbfRoC8SgQAvD_BwE) |
| Differential | [461.9](https://urun.n11.com/diger/diferansiyelli-on-arka-aks-mili-transmisyon-koprusu-110-P376769387?gclid=CjwKCAjwtqj2BRBYEiwAqfzur6cljI_MMaHoJzP-0aWy07Y1PWOxeJ6nng26h3QZGFltqoY6hM48rBoC8hMQAvD_BwE&gclsrc=aw.ds) |
| Total | 738.78 |

**5. Motion simulation**

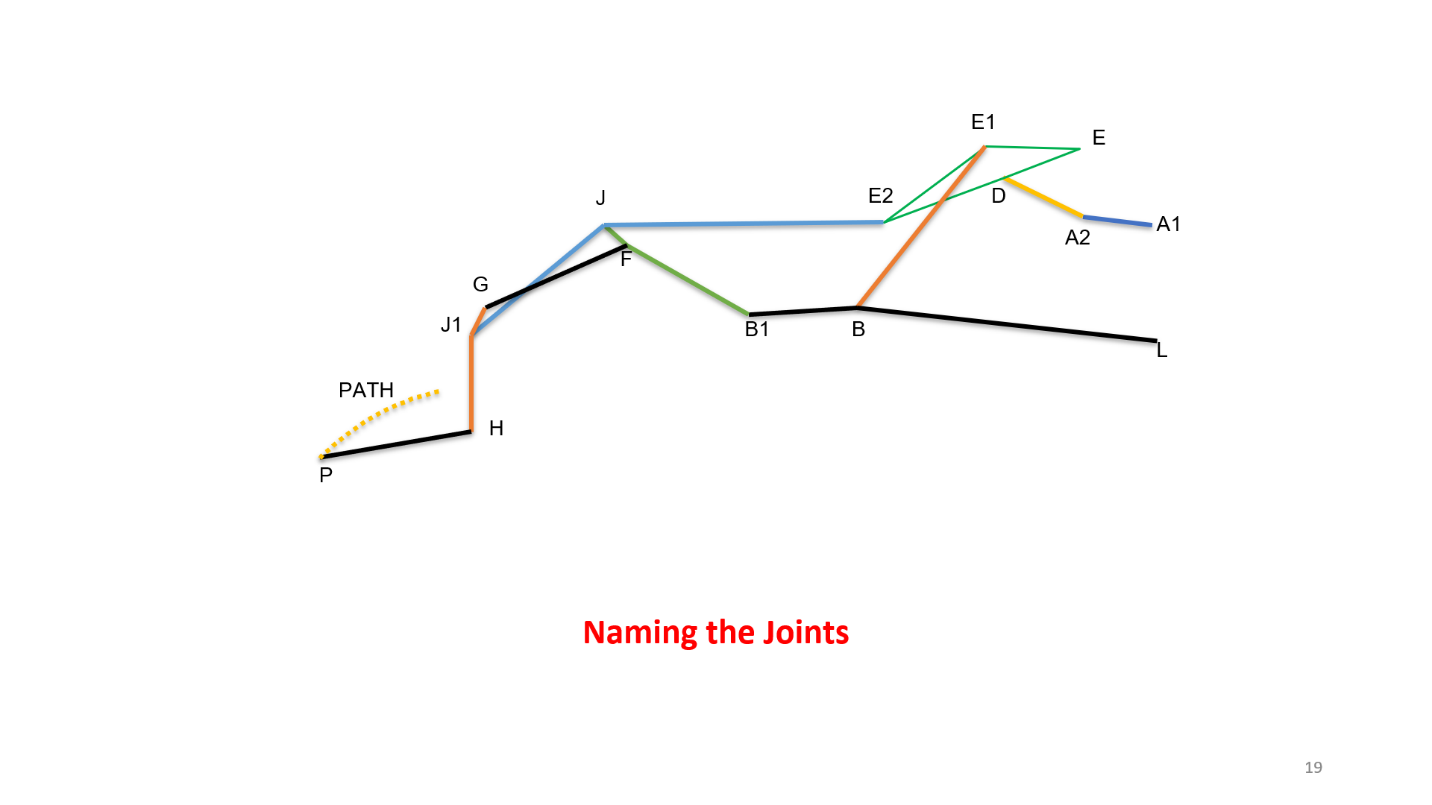
****

<https://drive.google.com/open?id=1BshLZDrfG4M0is3r9rgvGYumhSK9P294>

-We mainly use the concentricity as an constrain our system but the important constrain was the point on curve, on a slider joint to give the slider motion on a curve path.

-We run our system at 3pi rad/s because otherwise we could not get the full motion

**6. Derivation of Dynamic Equations**

****

**6.1.** **Planar Kinematics**

-For our equation:

****

**-Until Process:**

****-

-

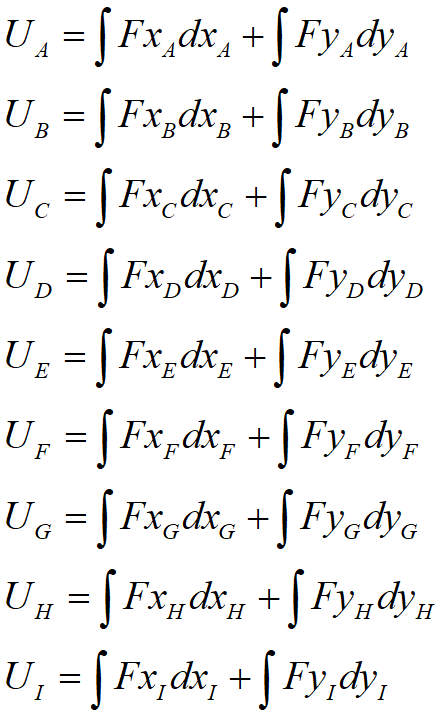
****

****

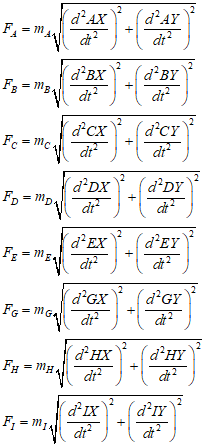
****

****

**-ENERGY EQUATİONS**

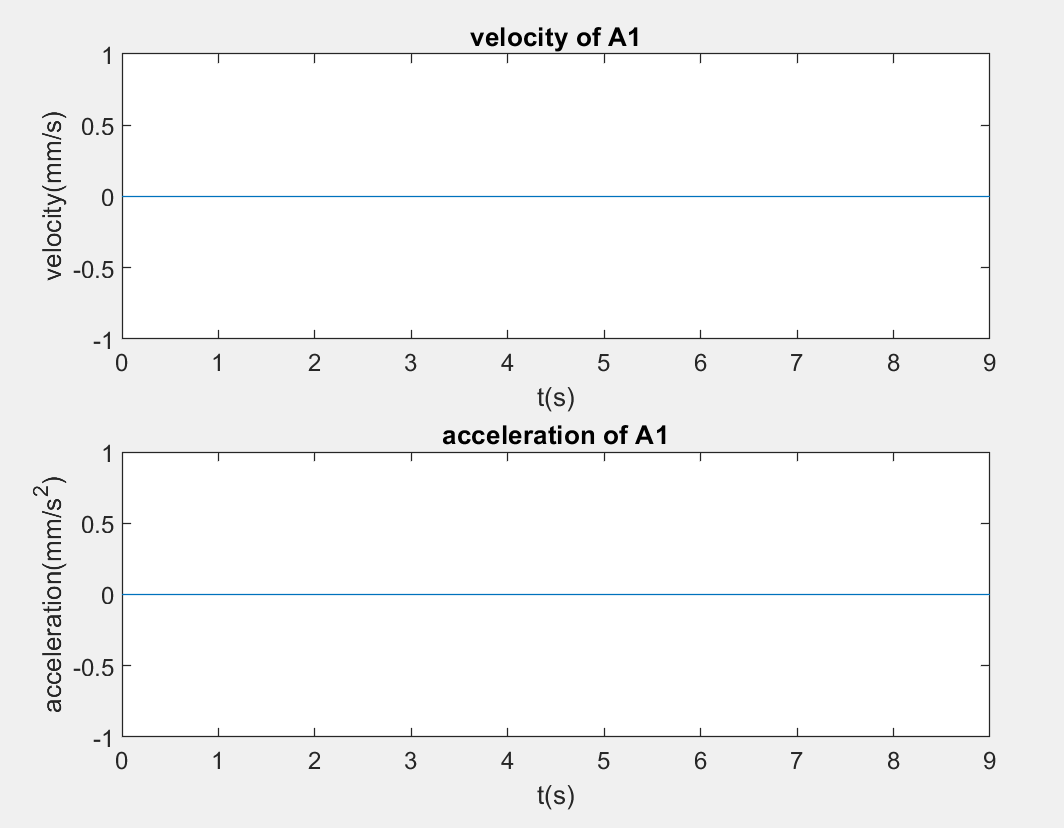


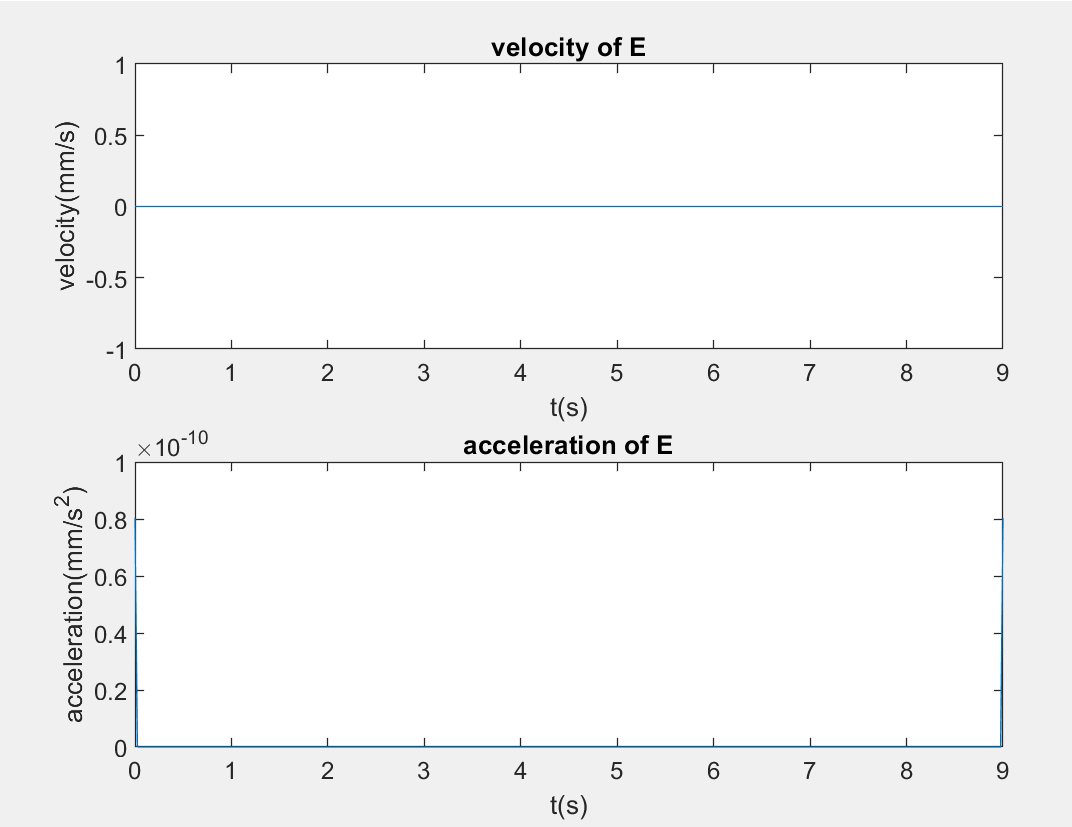
**-FORCE EQUATİONS:**

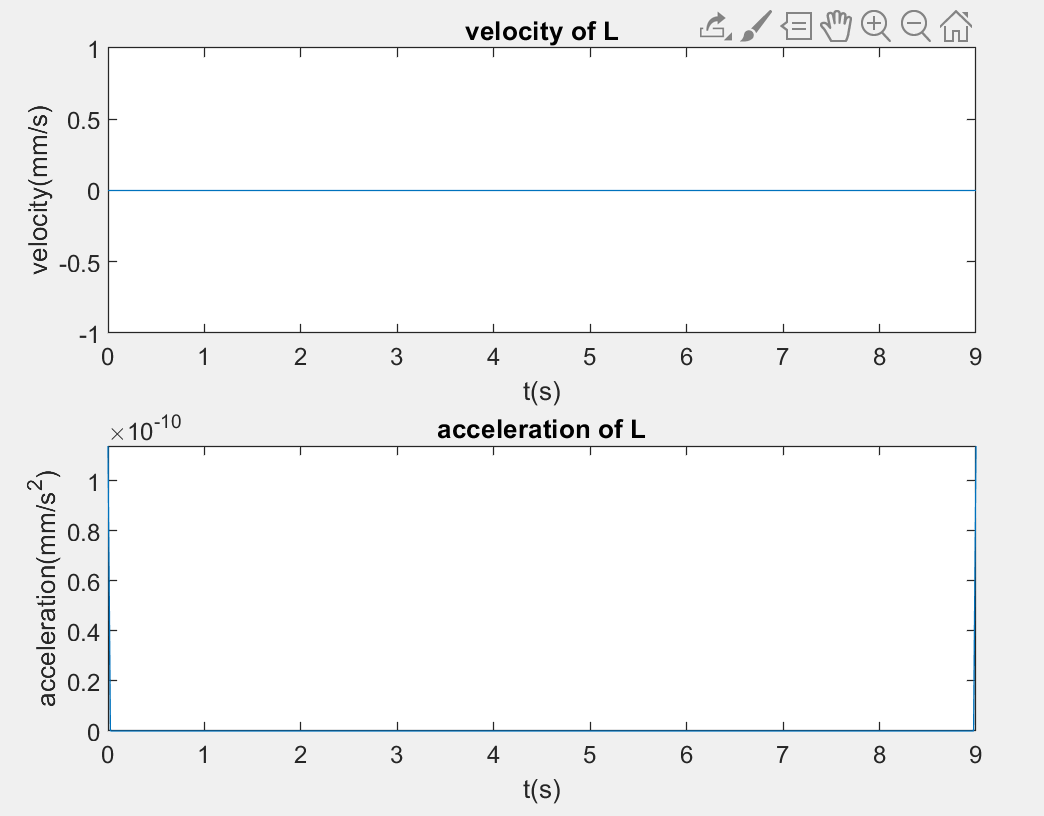
****

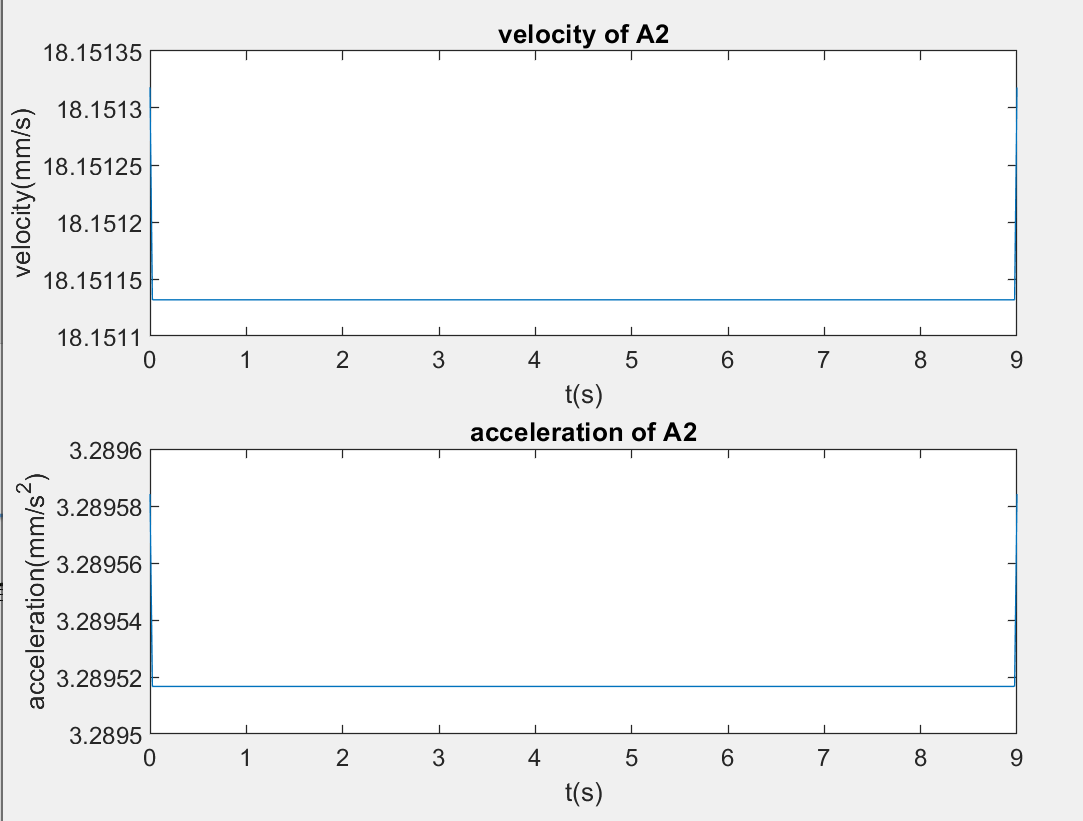
**7. MATLAB Analysis**

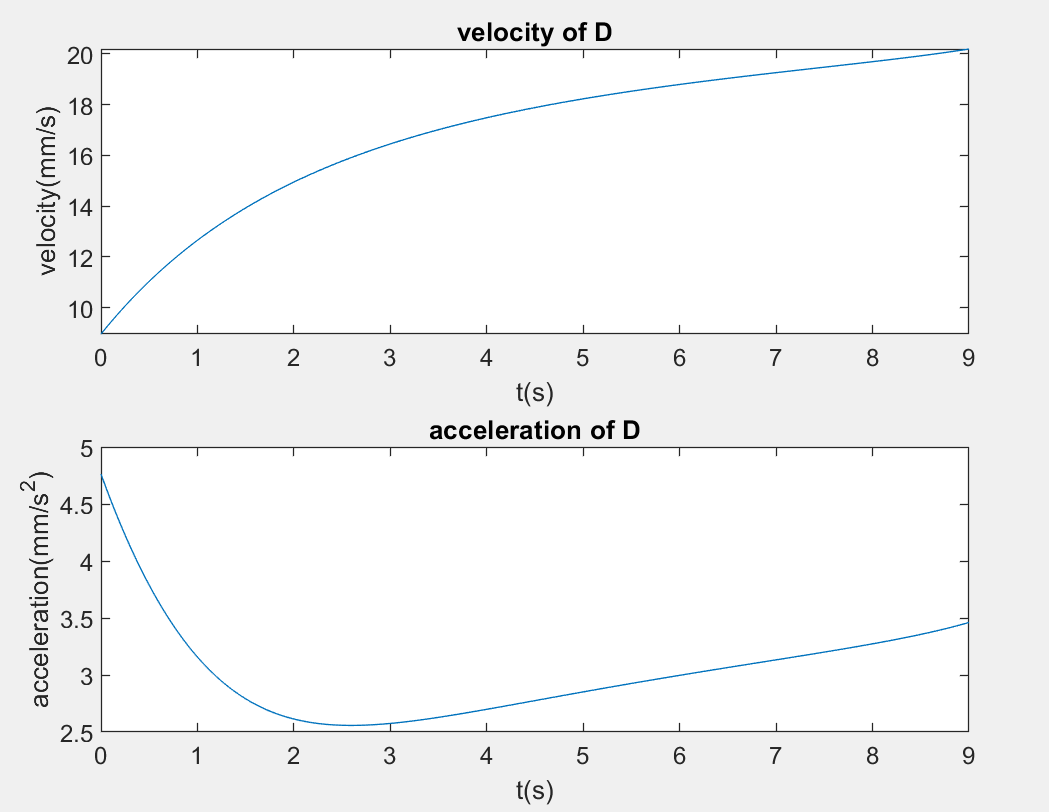
* **Plot of the equation of the velocity and accelerations of points:**

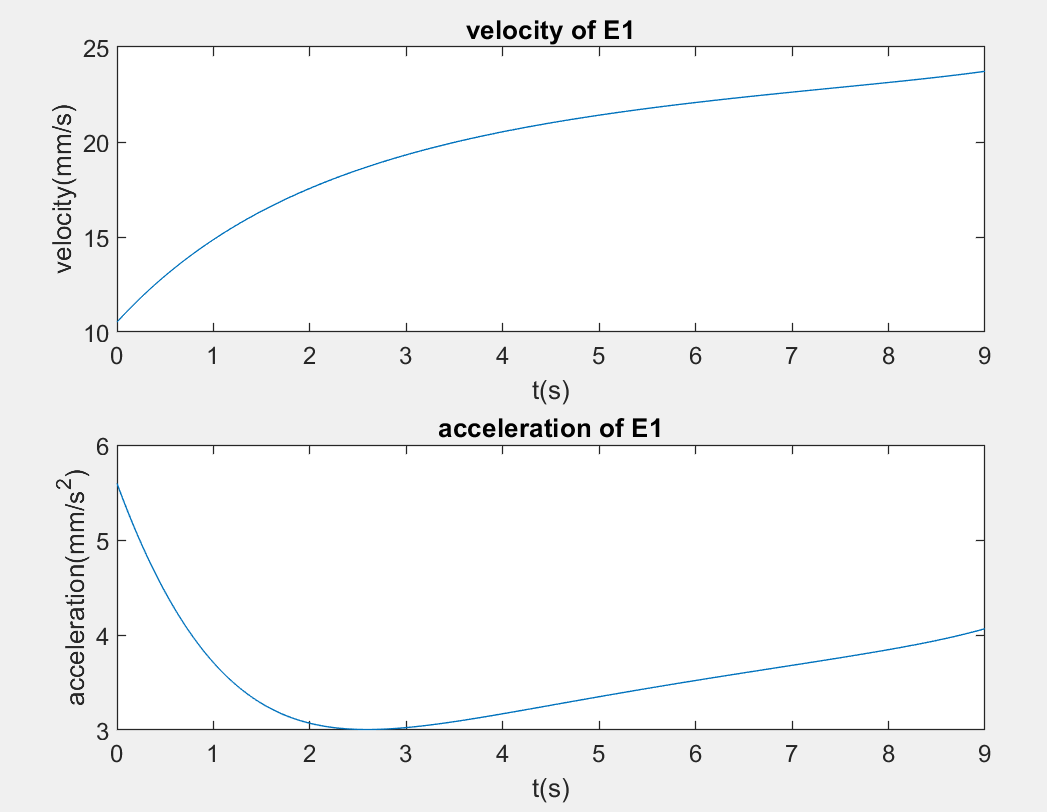


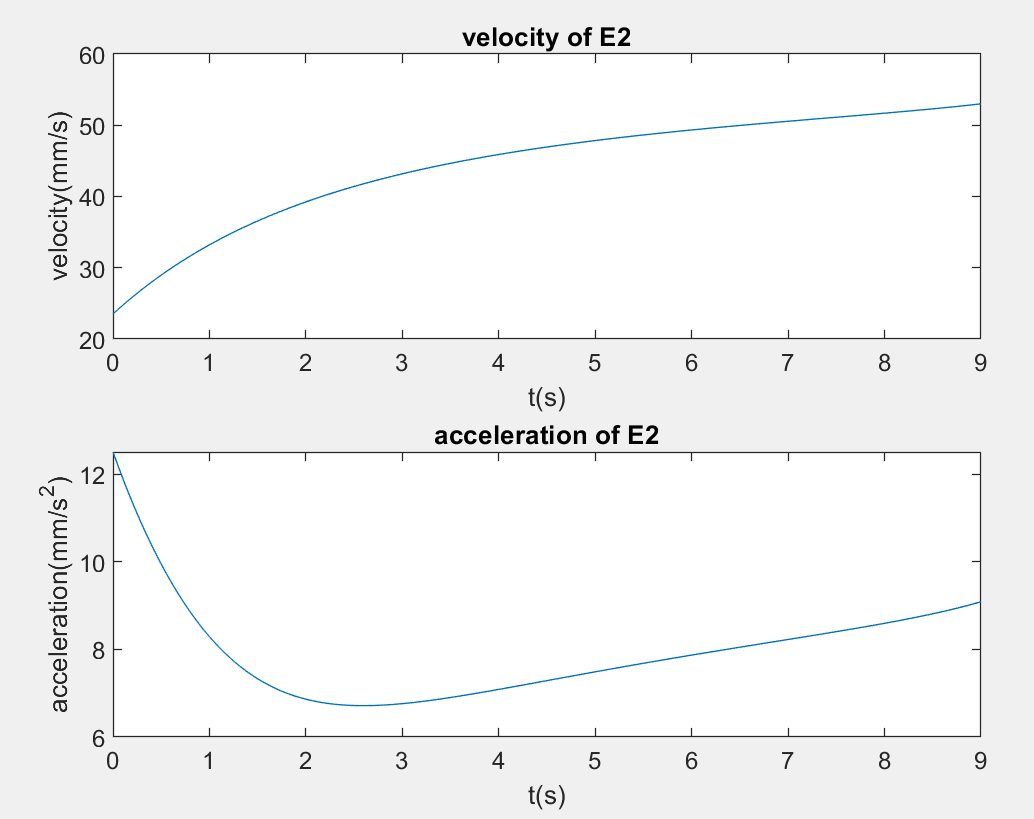


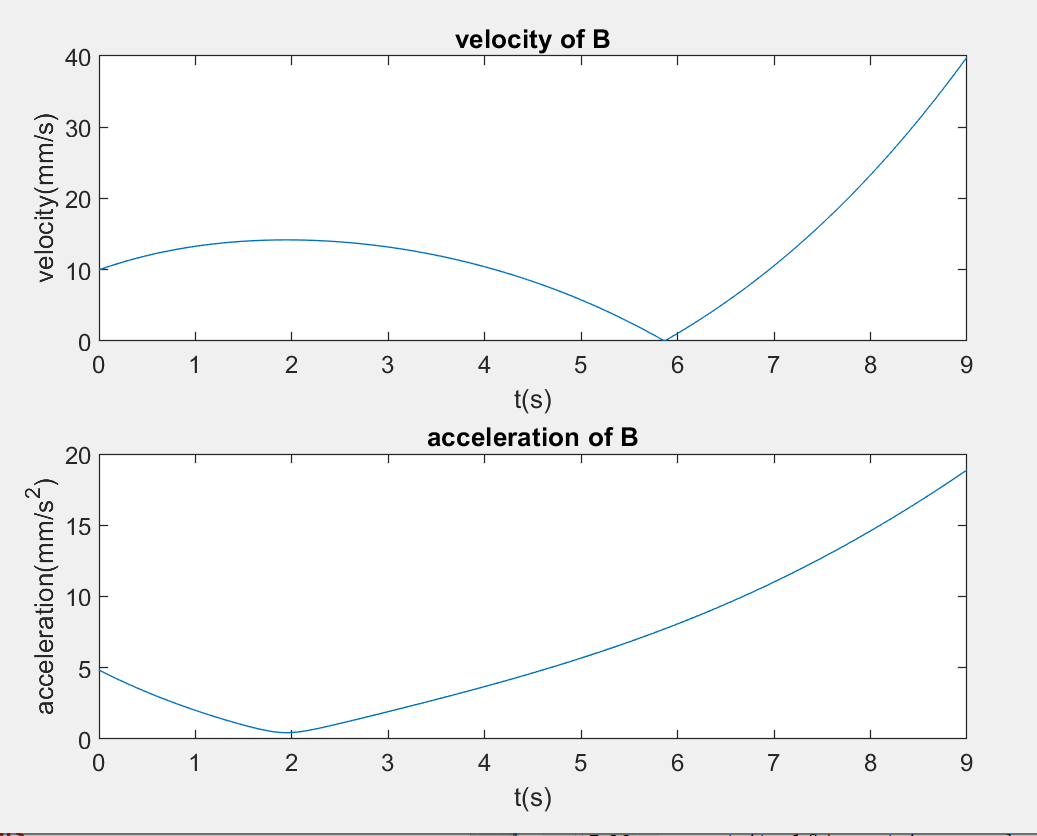


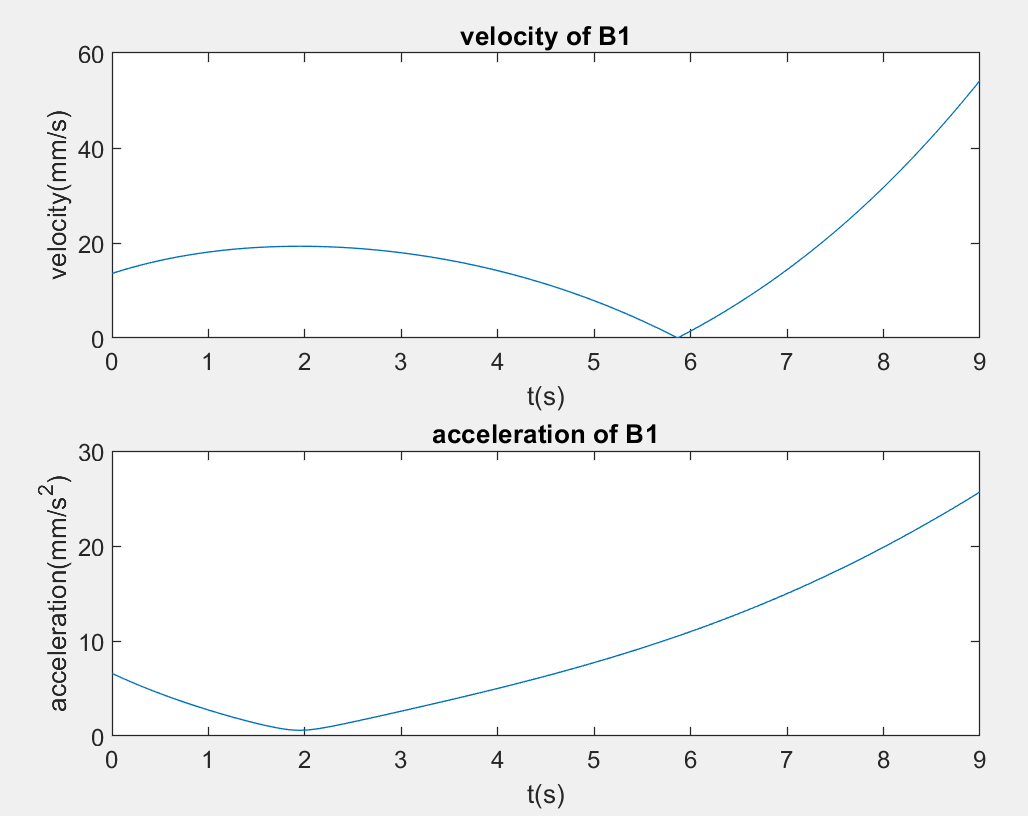


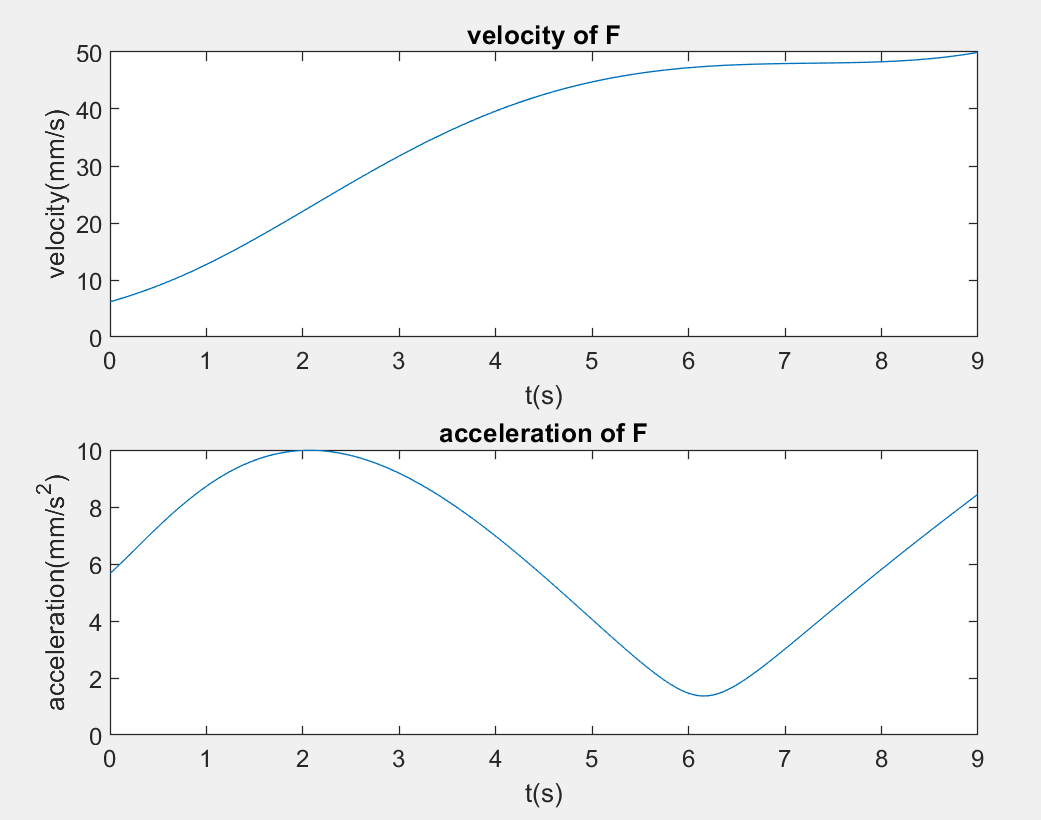


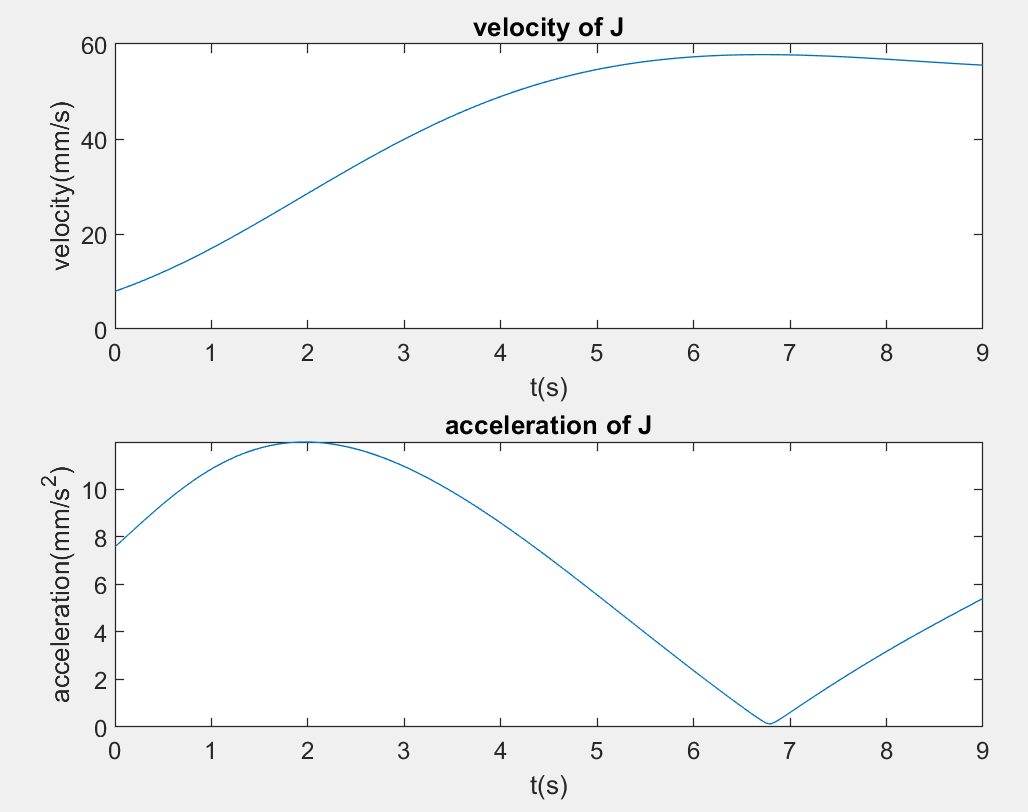


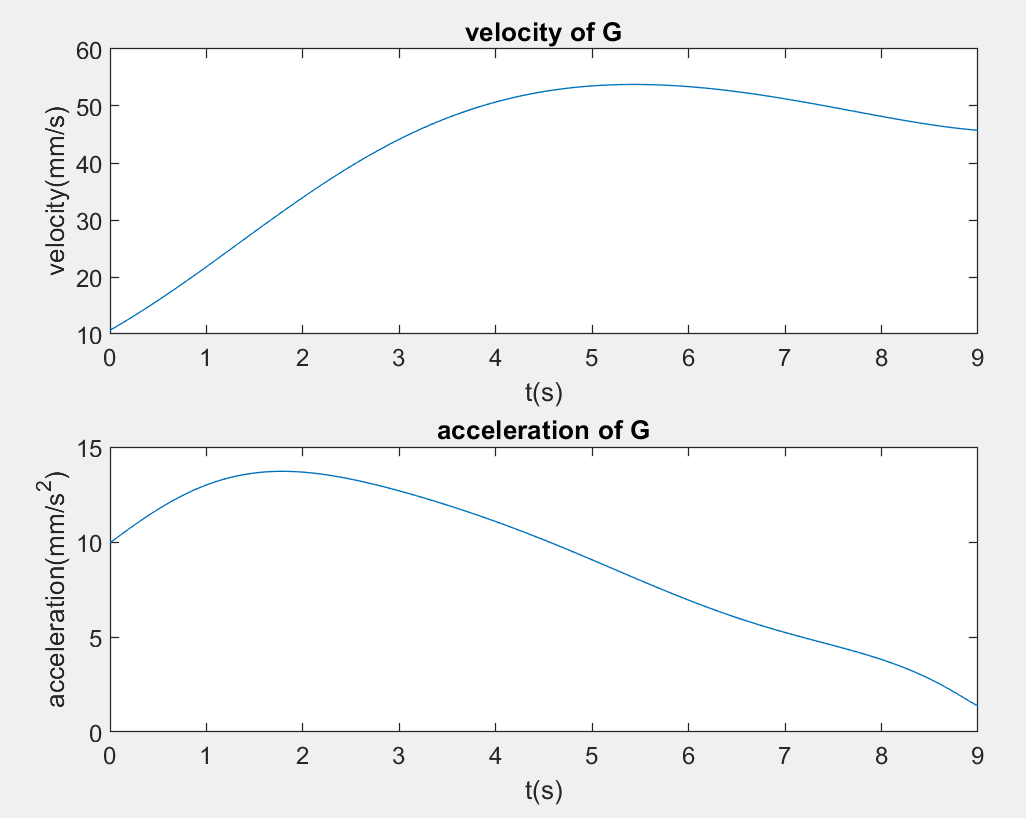


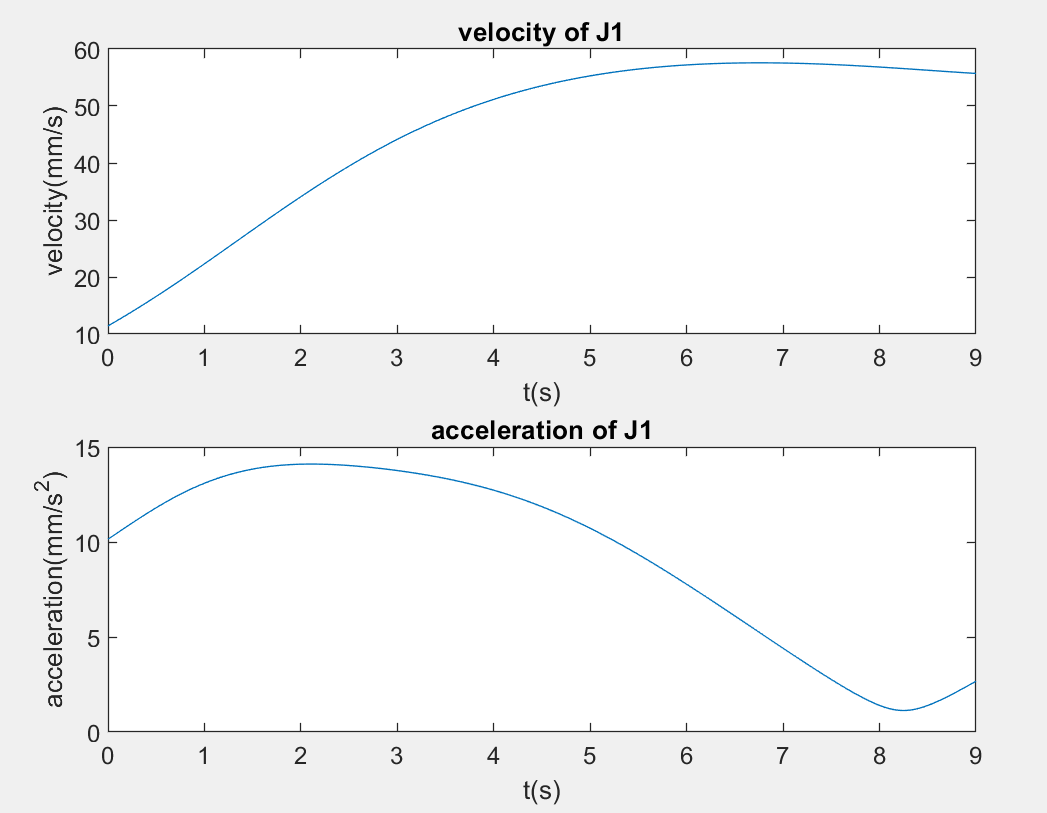


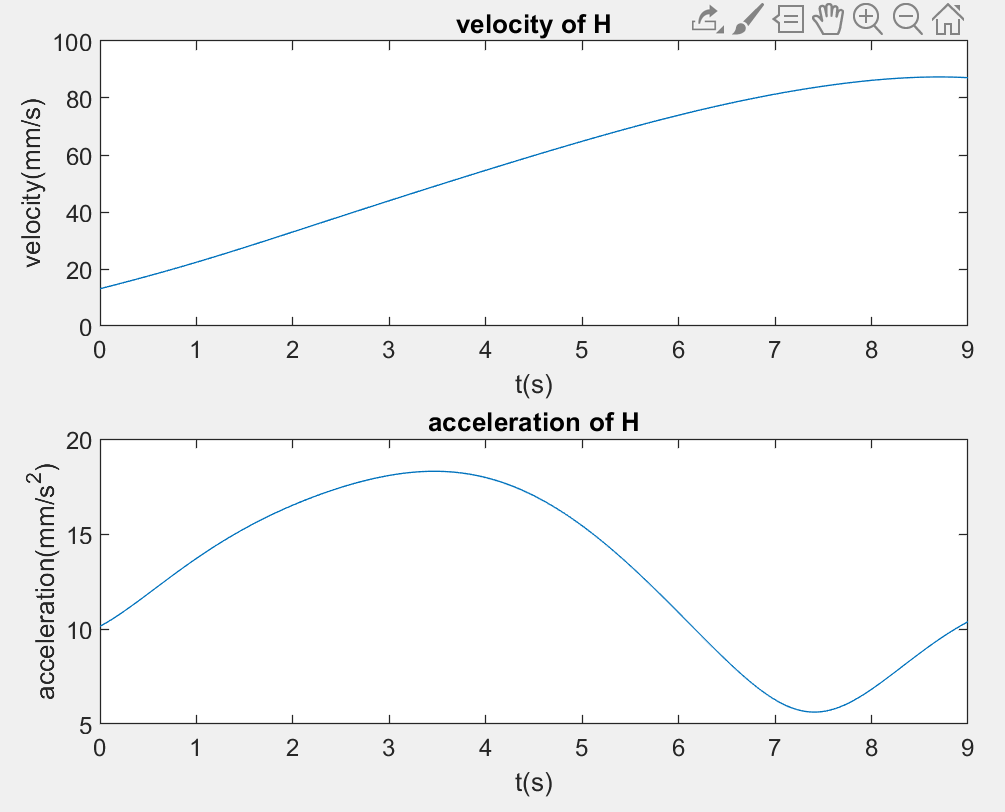


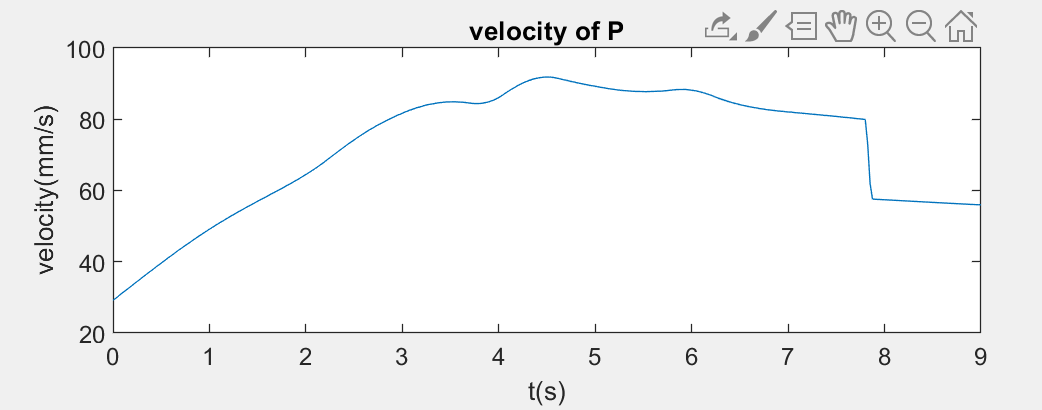




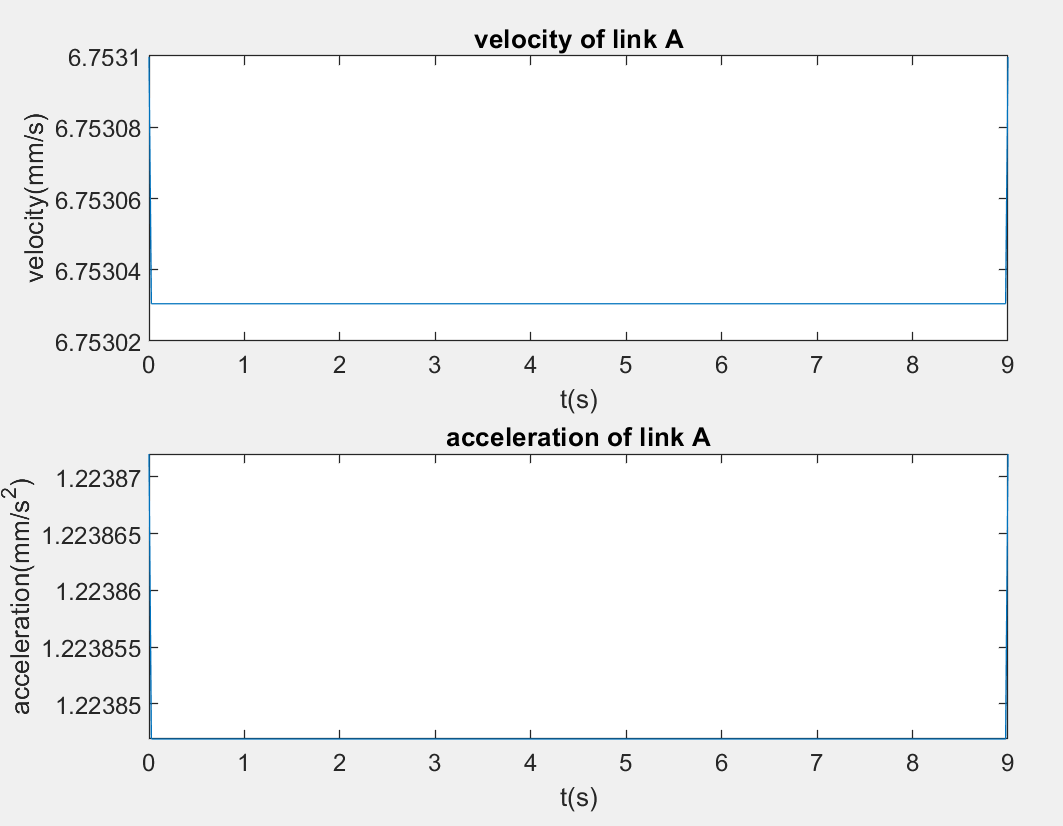


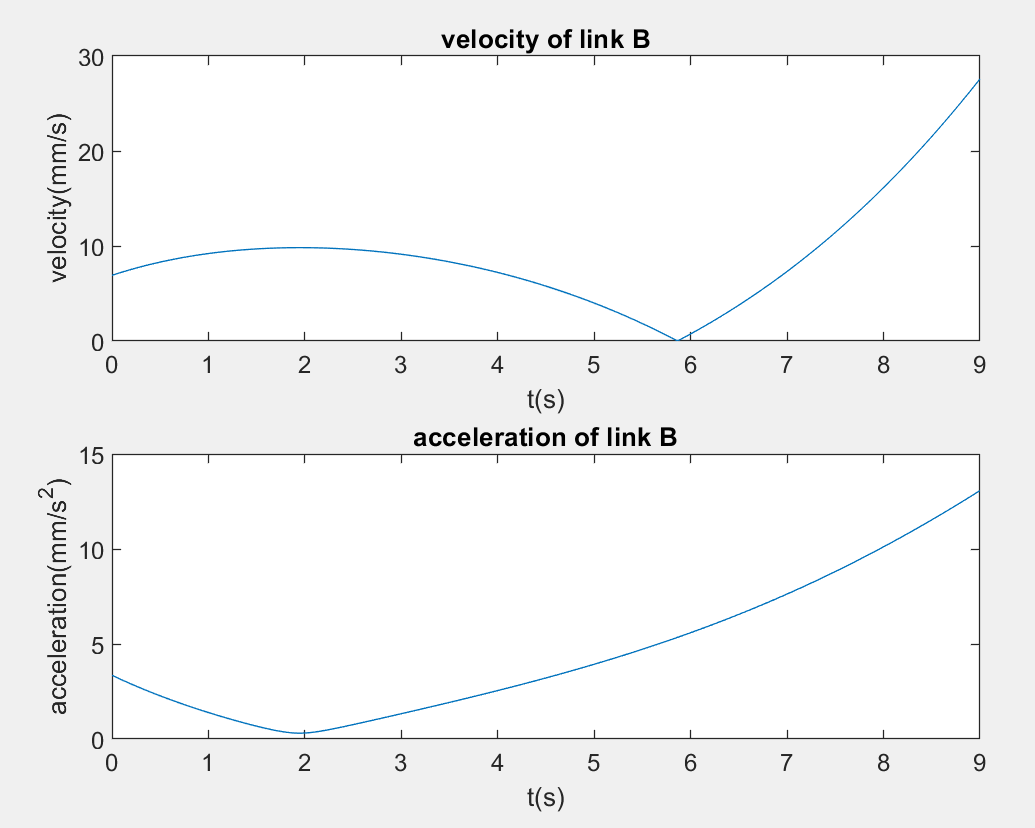


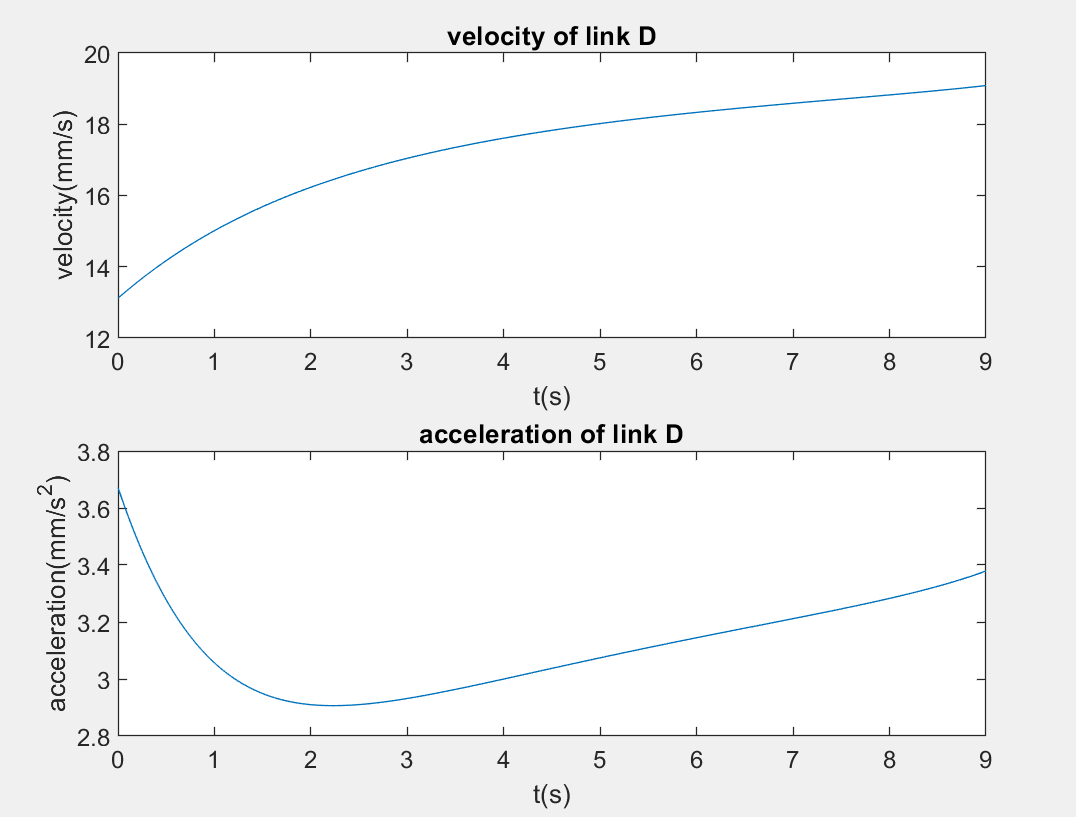


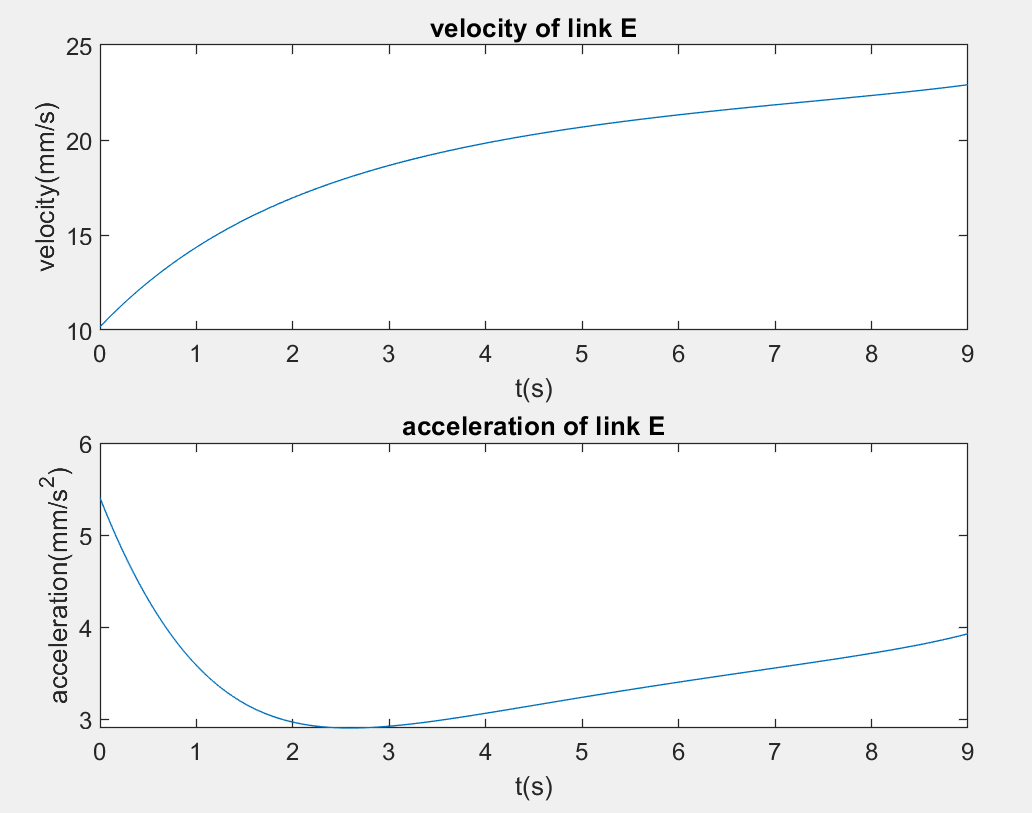


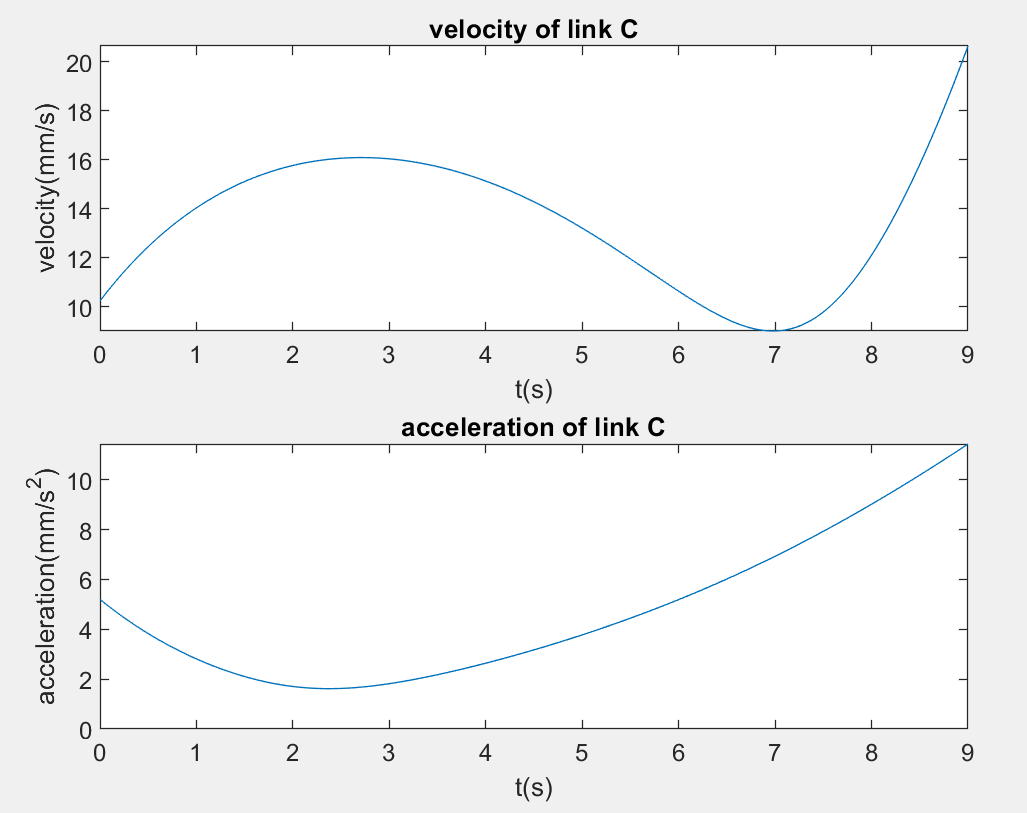
**-the plot of the equation of velocity and acceleration of links:**



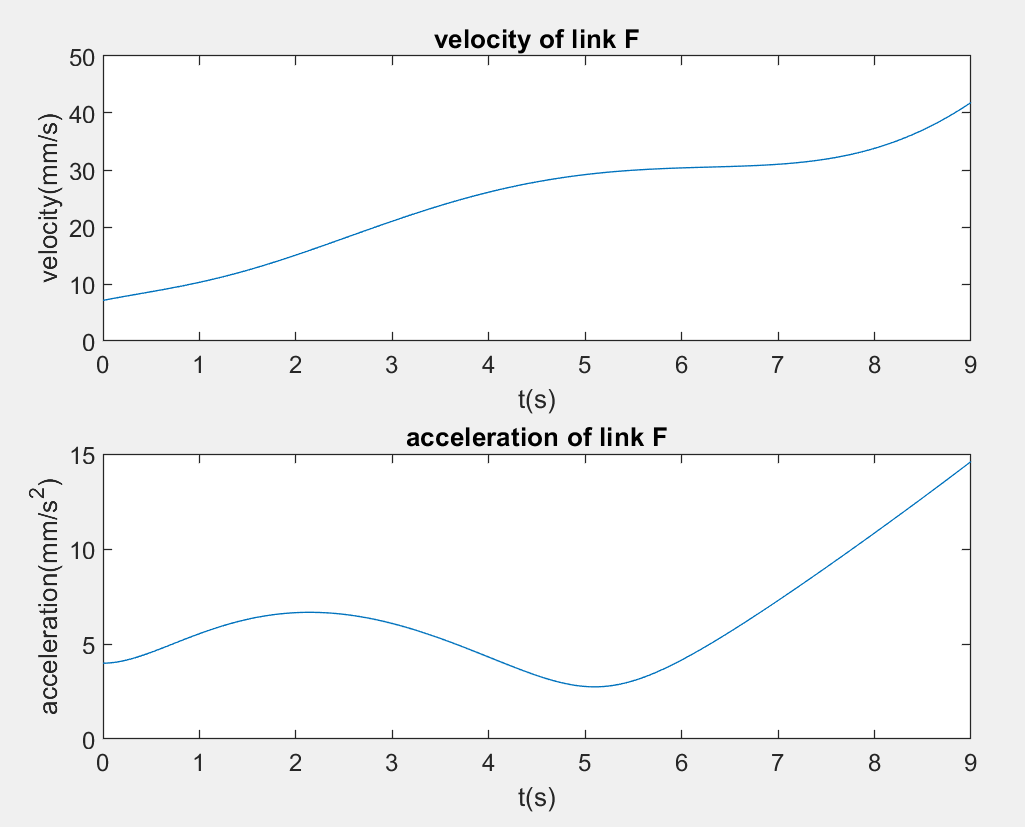


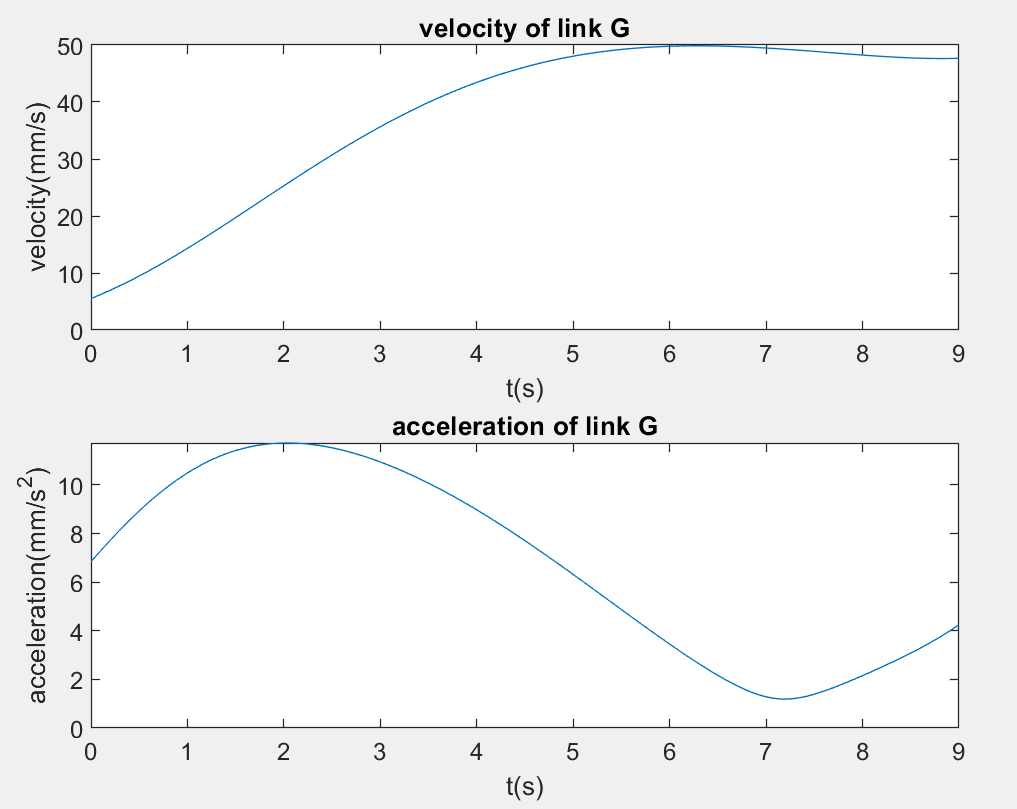


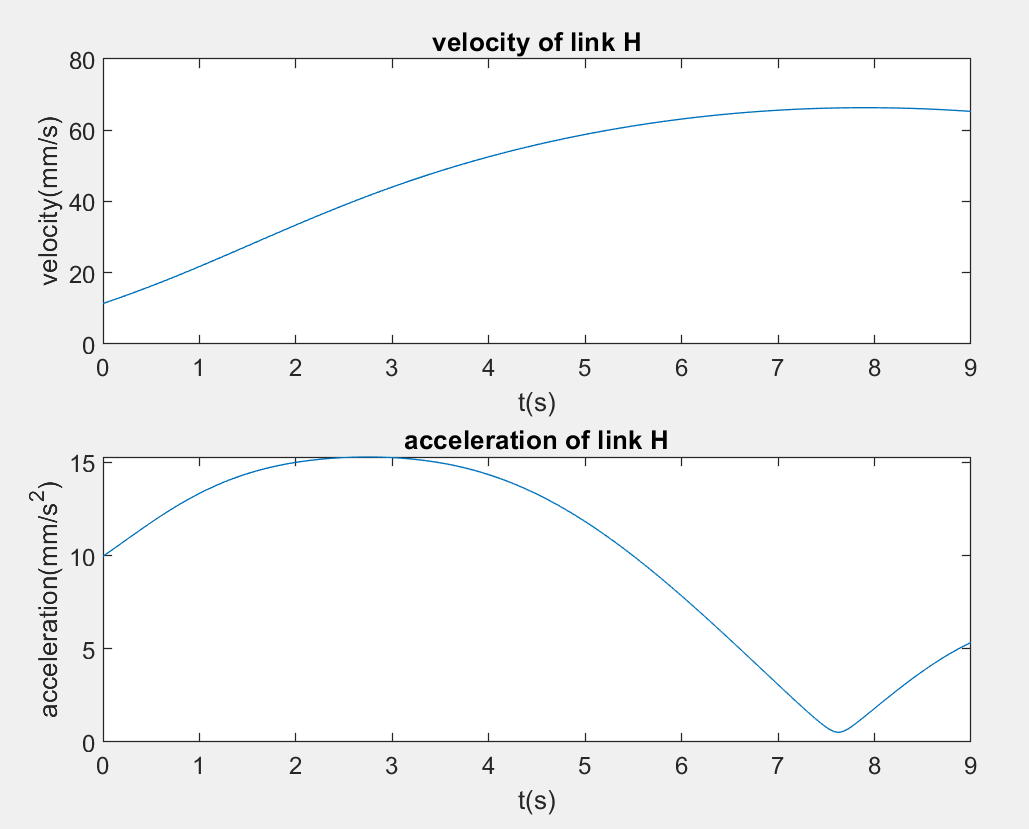


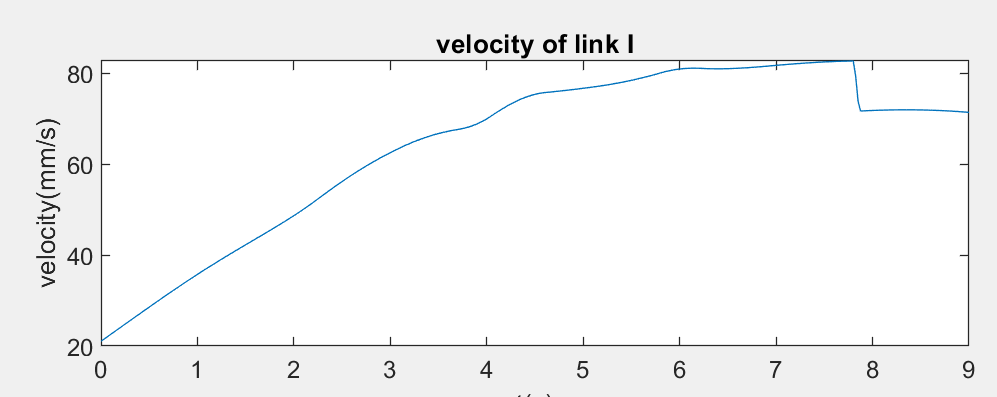


,

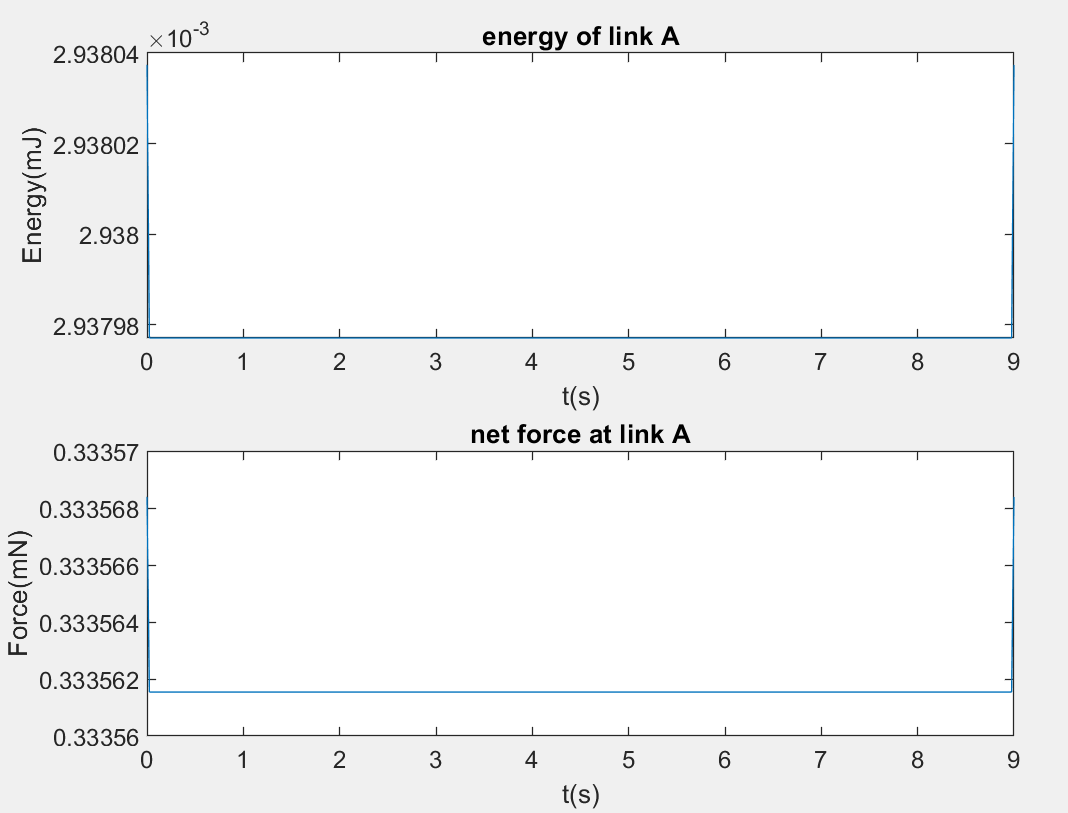


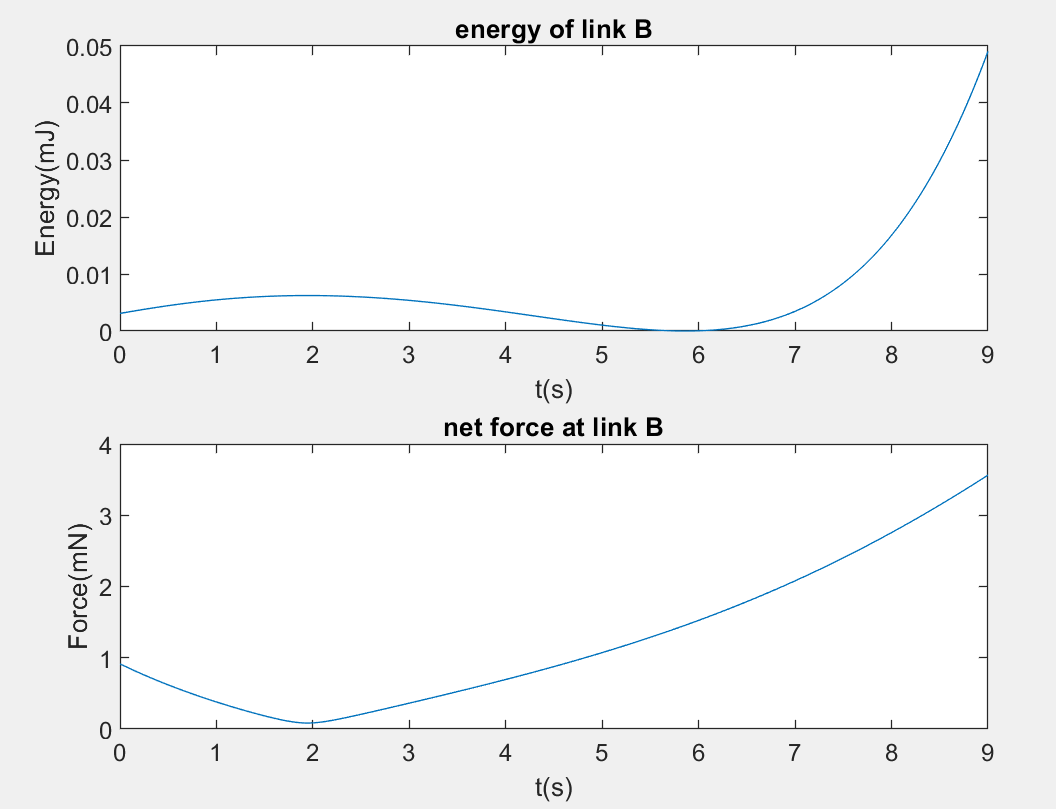


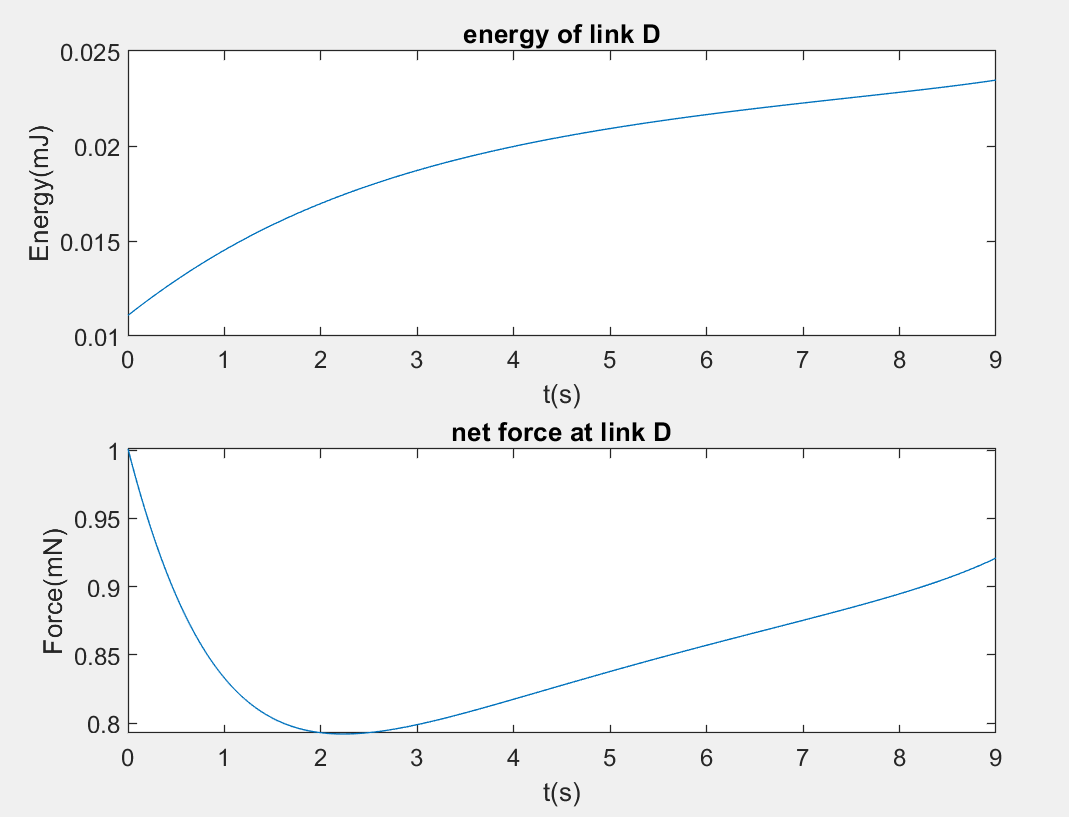


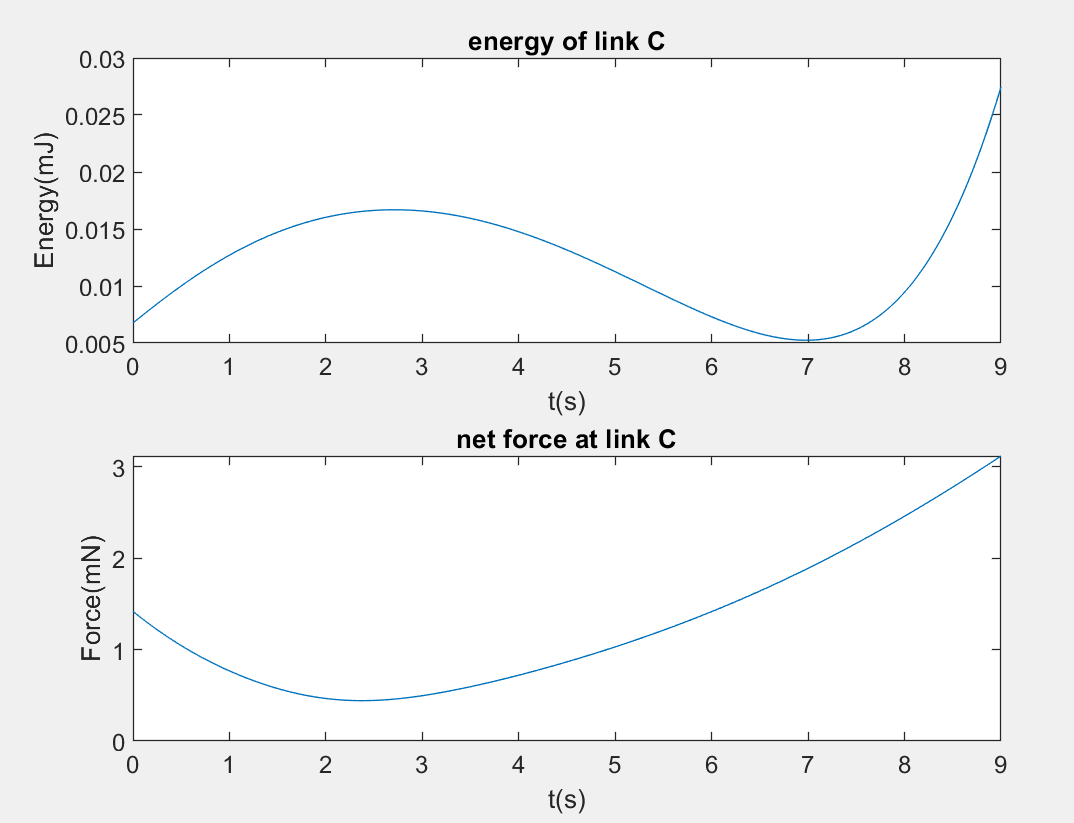


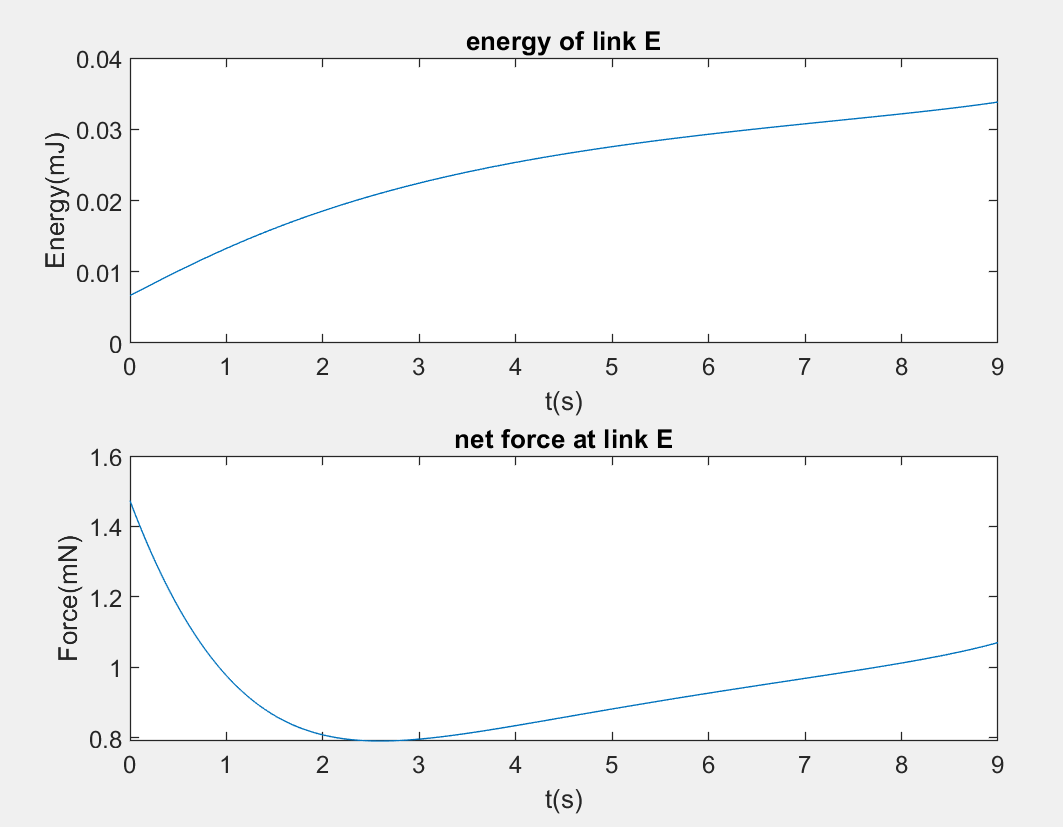
**-the plot of the equation of energy and force of the links:**

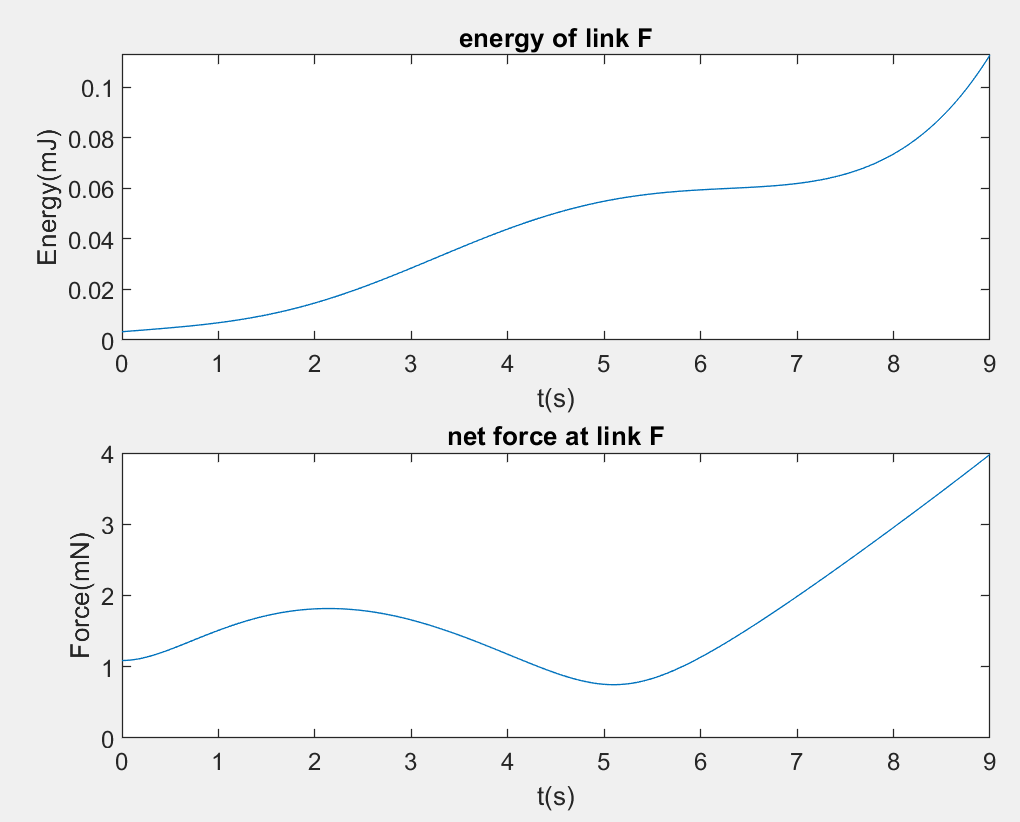


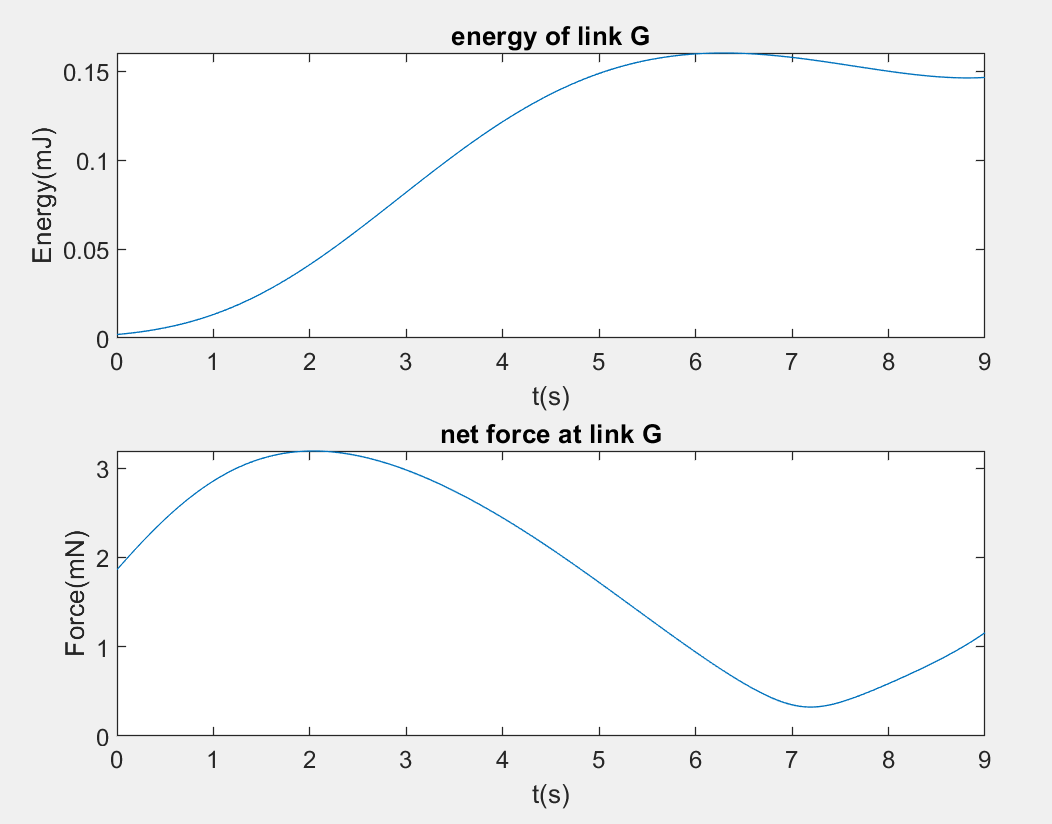


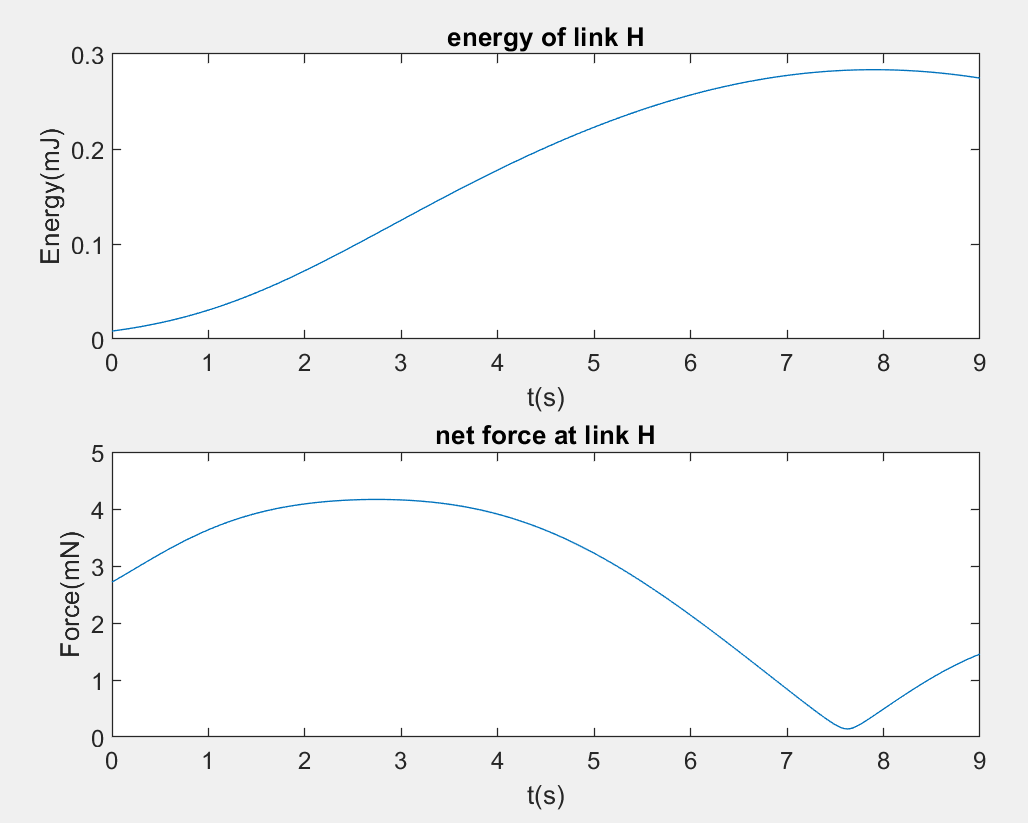






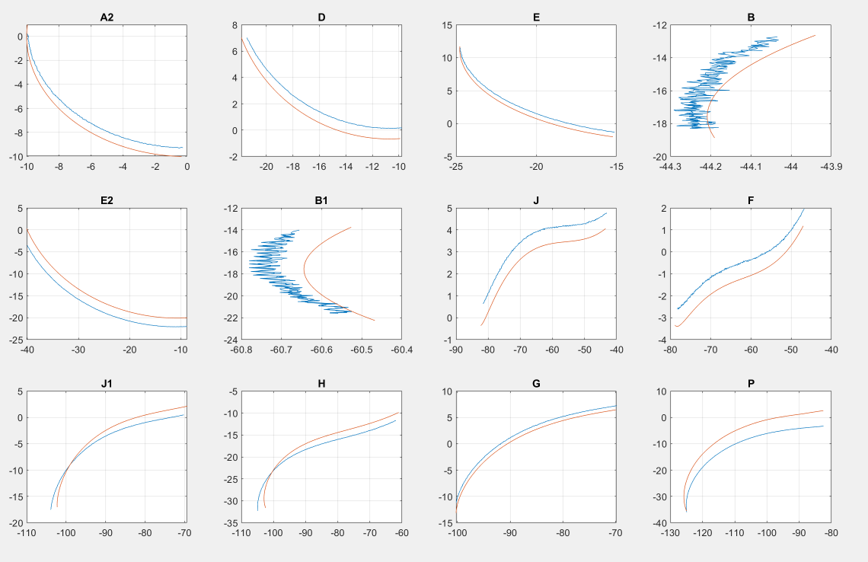






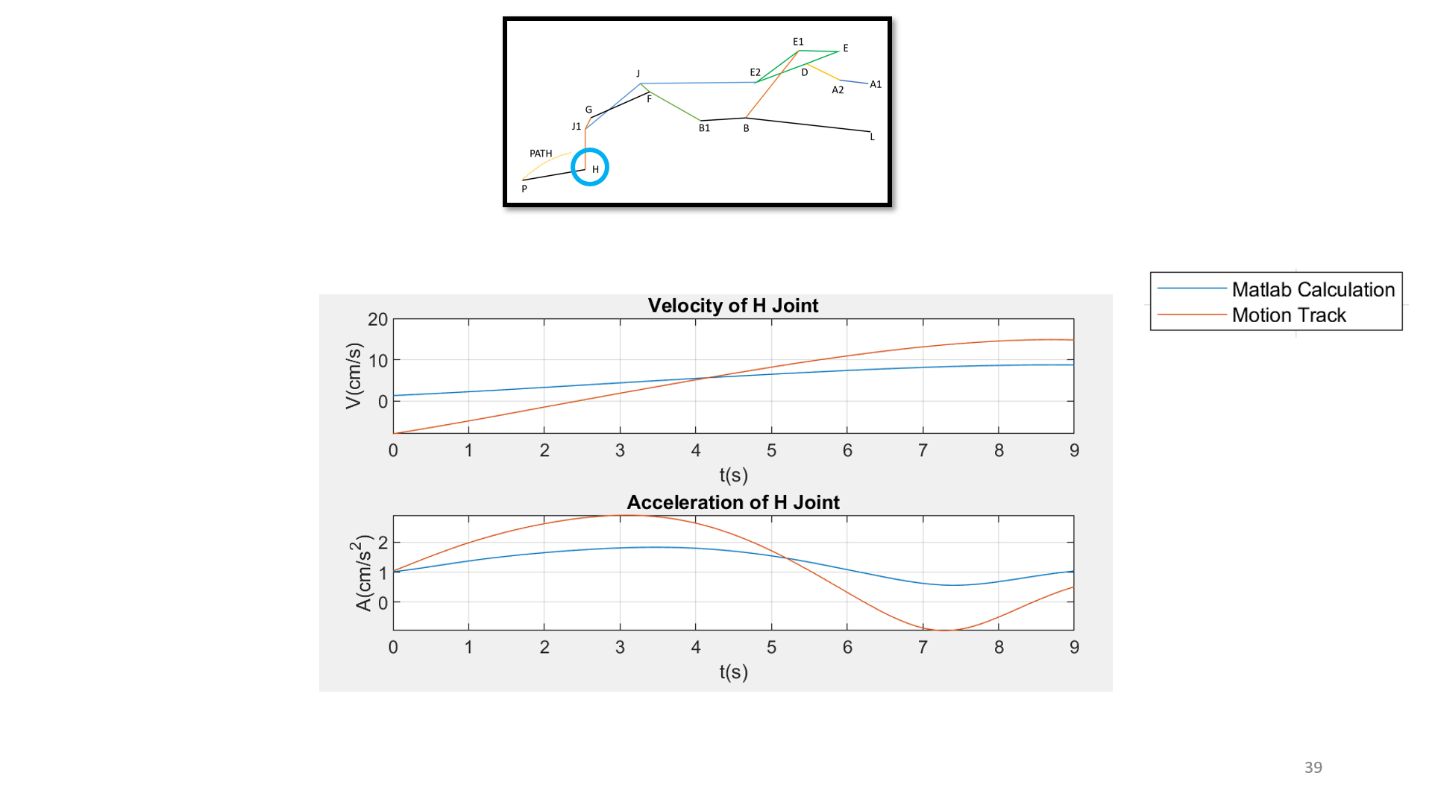
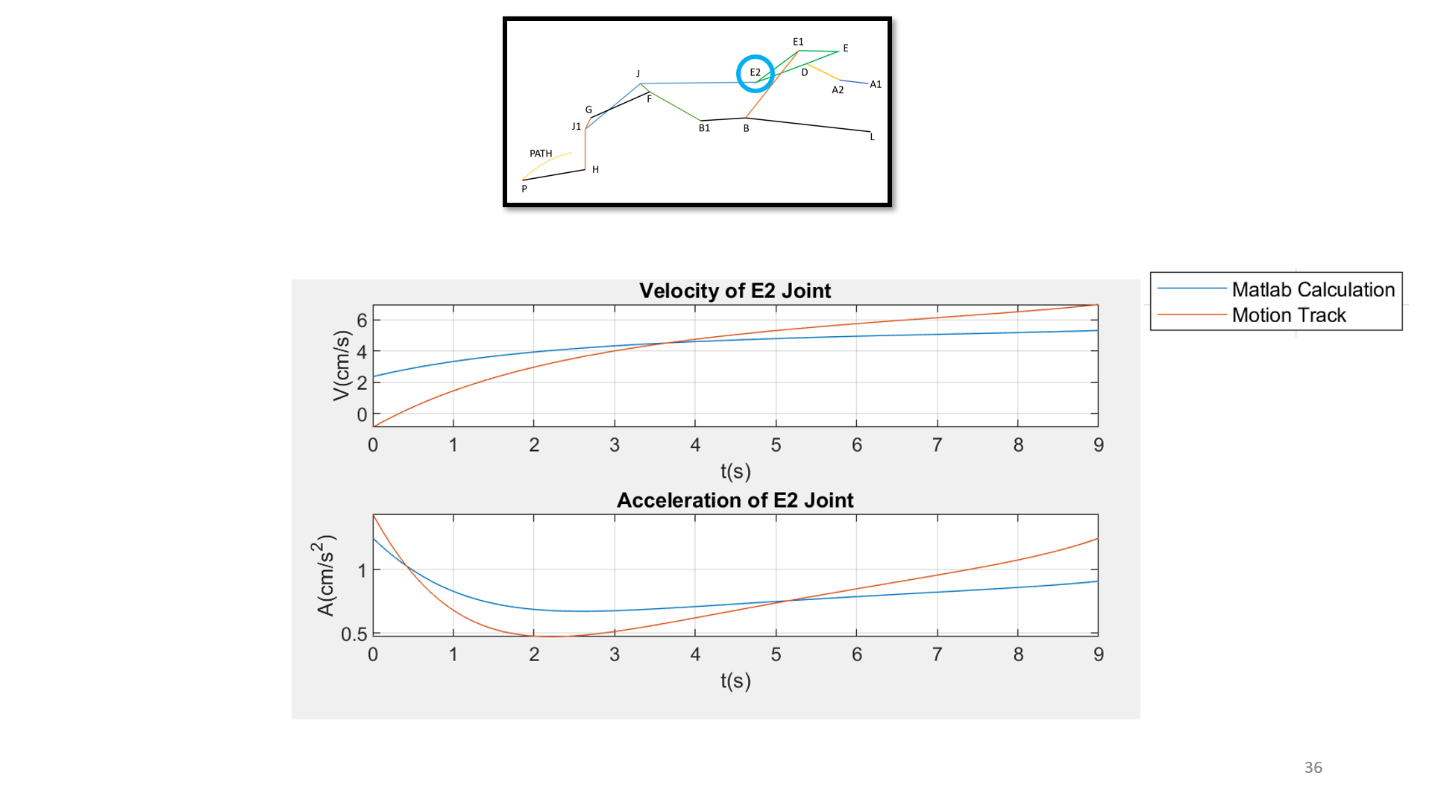
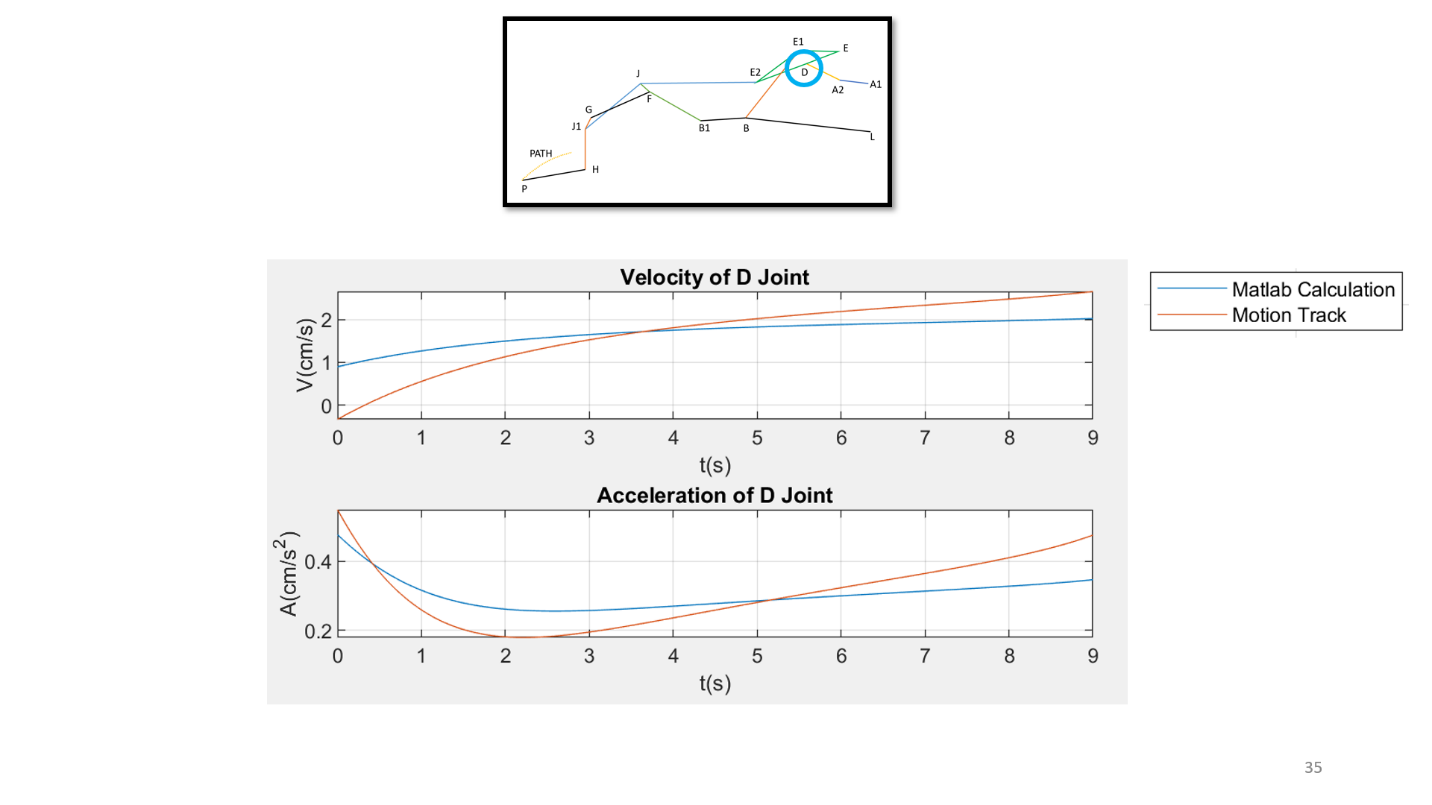
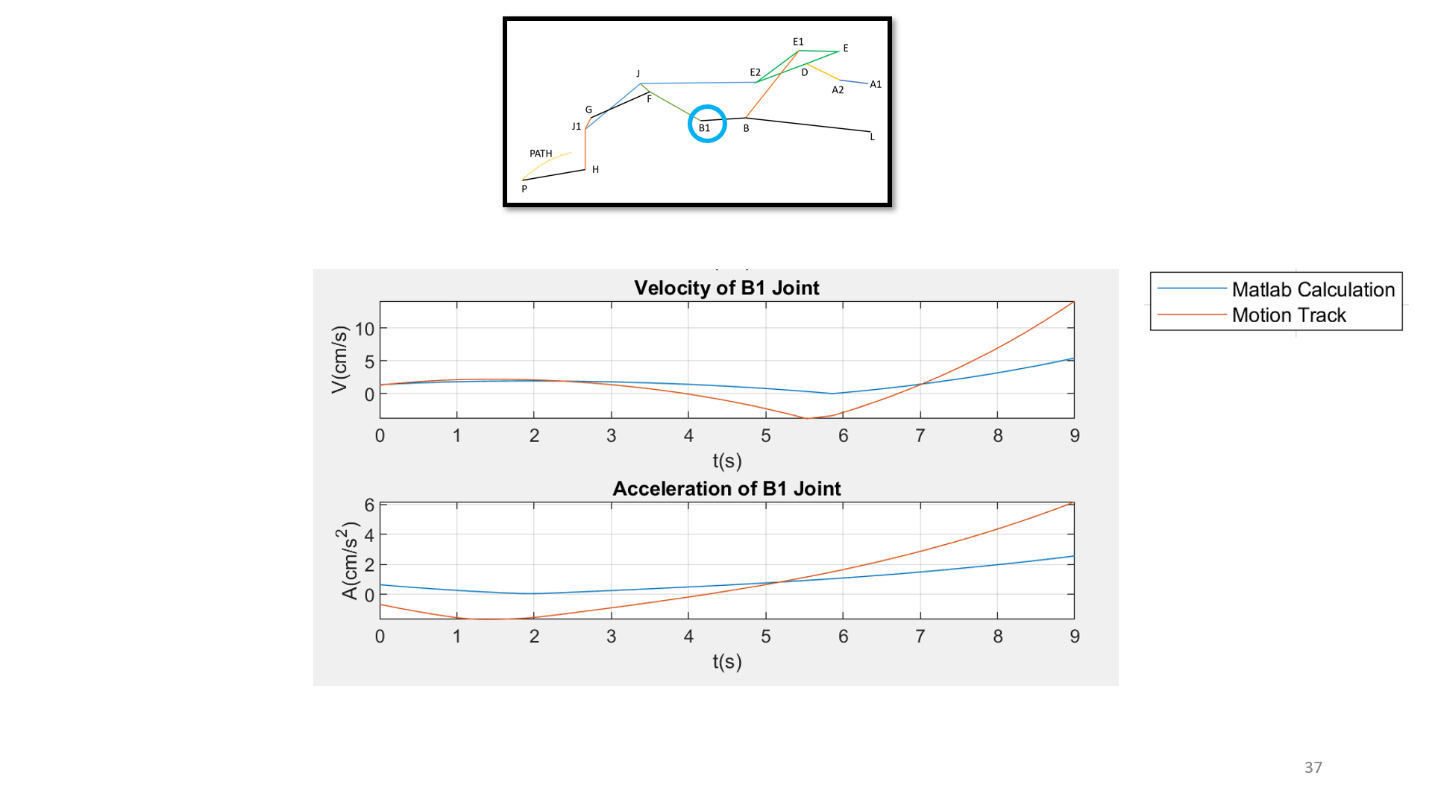
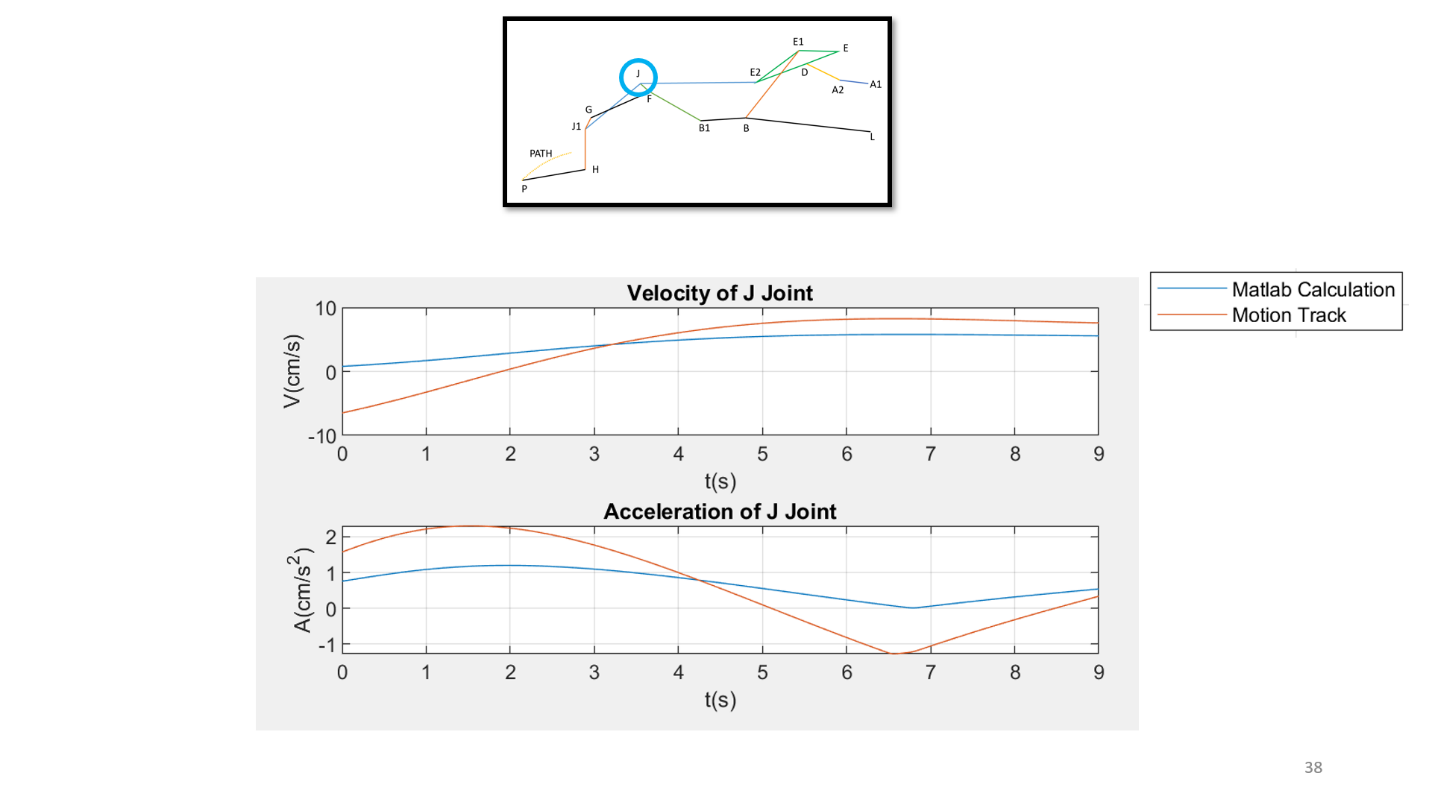
**8. Experimental Validation**

**Positions:**

****

The main differences between the positions is the number of arrays in the position as you see the path almost same expect B1 and B ( this is the because of pixels error when they are changing the color) and since we have a lot number of the value in array which come from the Matlab we can not see them as the same but if you look the G you can see when computer the catch lots of the data from the motion track we saw the almost perfect matching.

**Velocity and acceleration:**



-The main differences reason is the Nx essentially take calculating the values from its own numerical method and the to some program performances and the simulation errors it didn’t gives the perfect calculation and our data also not perfect because we are not considering the any friction or the motor performance and the turbulence effects of the plane on the system we think that everything happen in perfectly idealized environment.

**9. Conclusion**

We learned the theoretical value never gives the exact value for a real system application even when find the equations both numerical and analytical and the from computer they were not giving the same results. So, whatever we find the theoretical value we must always also evaluate the real system values. We also find the changes to practice what we learn during the lecture and its application on the real life as an engineer. We also learn to how presentation skills are important. We learned that real life is not easy on the aerospace area is not easy.

**10. References**

[1] Rudolph, Peter K. C., NASA, “Mechanical Design of High Lift Systems for High Aspect Ratio Swept Wings”,1998

<https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19980021287.pdf>

[2] Hall, Nancy, Nasa Glenn Research Center, “The Lift Equation”, 2015

<https://www.grc.nasa.gov/WWW/K-12/airplane/lifteq.html>

[3] Lopez, Daniel R., NASA, “Flaps and Slats”, 2018

<https://www.grc.nasa.gov/WWW/K-12/airplane/flap.html>

**11. Appendix**

**Matlab Codes:**

**-To track the points;**

% track\_point.m

clc

clear

close all

%% Get video

vid = VideoReader('mech206motion1.avi'); % Load video

nFrames = vid.NumberOfFrames; % Get number of frames

M = zeros(nFrames,2); % Matrix A to store x and y position data

%% Tracking point color values

red = 0 ; % Red color value

green =0; % Green color value

blue = 248; % Blue color value

range = 20; % Range used to add/subtract from color values

%% Track point at each frame

for k=1:nFrames

a = read(vid,k); % Read current frame

fr = a(:,:,1);

fg = a(:,:,2);

fb = a(:,:,3);

%% Get the pixels in given color range

P = ((red+range>fr) & (fr>red-range) &...

(green+range>fg) & (fg>green-range) &...

(blue+range>fb) & (fb>blue-range));

[y1,x1] = find(P==1);

Px = mean(x1); % Take the mean to get the center

Py = mean(y1);

M(k,1) = Px; % Store point in matrix A

M(k,2) = Py;

%% Plot figure

imshow(a)

hold on

title(['Step ',num2str(k),': X = ',num2str(Px),' Y = ',num2str(Py)])

plot(Px,Py,'ro','linewidth',2)

pause(0.01)

end

%% Plot position data

A\_mm = M \* 0.34; % Position data converted from pixels to millimeters. Change this

%value according to your scale.

% Plot raw position data

figure(); hold on;

title('Raw Tracking Data');

xlabel('X [mm]');

ylabel('Y [mm]');

plot(A\_mm(:,1),A\_mm(:,2))

% Filter data

xf = smooth(A\_mm(:,1)); % Smooths data using a 5-point moving average.

yf = smooth(A\_mm(:,2));

% Plot filtered data

figure(); hold on;

title('Filtered Position Data');

xlabel('X [mm]');

ylabel('Y [mm]');

plot(xf,yf)

disp(M)

%

% vid = VideoReader('mech206motion1.avi'); % Load video

% nFrames = vid.NumberOfFrames; % Get number of frames

% K = zeros(nFrames,2); % Matrix A to store x and y position data

% %% Tracking point color values

% red = 3 ; % Red color value

% green = 4; % Green color value

% blue = 255; % Blue color value

% range = 20; % Range used to add/subtract from color values

% %% Track point at each frame

% for k=1:nFrames

% a = read(vid,k); % Read current frame

% fr = a(:,:,1);

% fg = a(:,:,2);

% fb = a(:,:,3);

% %% Get the pixels in given color range

% P = ((red+range>fr) & (fr>red-range) &...

% (green+range>fg) & (fg>green-range) &...

% (blue+range>fb) & (fb>blue-range));

% [y1,x1] = find(P==1);

% Px = mean(x1); % Take the mean to get the center

% Py = mean(y1);

% K(k,1) = Px; % Store point in matrix A

% K(k,2) = Py;

% %% Plot figure

% imshow(a)

% hold on

% title(['Step ',num2str(k),': X = ',num2str(Px),' Y = ',num2str(Py)])

% plot(Px,Py,'ro','linewidth',2)

% pause(0.01)

% end

% %% Plot position data

% A\_mm = K \* 0.34; % Position data converted from pixels to millimeters. Change this

% %value according to your scale.

% % Plot raw position data

% figure(); hold on;

% title('Raw Tracking Data');

% xlabel('X [mm]');

% ylabel('Y [mm]');

% plot(A\_mm(:,1),A\_mm(:,2))

% % Filter data

% xf = smooth(A\_mm(:,1)); % Smooths data using a 5-point moving average.

% yf = smooth(A\_mm(:,2));

% % Plot filtered data

% figure(); hold on;

% title('Filtered Position Data');

% xlabel('X [mm]');

% ylabel('Y [mm]');

% plot(xf,yf)

%

% disp(K)

**-Velocity data from which we give you in drive:**

function importfile(fileToRead1)

%IMPORTFILE(FILETOREAD1)

% Imports data from the specified file

% FILETOREAD1: file to read

% Auto-generated by MATLAB on 25-May-2020 14:31:17

% Import the file

newData1 = load('-mat', fileToRead1);

% Create new variables in the base workspace from those fields.

vars = fieldnames(newData1);

for i = 1:length(vars)

assignin('base', vars{i}, newData1.(vars{i}));

end

**-Acceleration data from which we give you in drive:**

function importfile(fileToRead1)

%IMPORTFILE(FILETOREAD1)

% Imports data from the specified file

% FILETOREAD1: file to read

% Auto-generated by MATLAB on 25-May-2020 14:33:35

% Import the file

newData1 = load('-mat', fileToRead1);

% Create new variables in the base workspace from those fields.

vars = fieldnames(newData1);

for i = 1:length(vars)

assignin('base', vars{i}, newData1.(vars{i}));

end

**-To first derivative to the velocity and acceleration:**

function a = first\_derivative(x,dt)

N=length(x);

a(1)=(-x(3)+4\*x(2)-3\*x(1))/2/dt;

a(N)=-(-x(N-2)+4\*x(N-1)-3\*x(N))/2/dt;

for i=2:N-1

a(i)=(x(i+1)-x(i-1))/2/dt;

end

end

**-To second derivative:**

function a = second\_derivative(x,dt)

N=length(x);

a(1)=(-x(4)+4\*x(3)-5\*x(2)+2\*x(1))/dt^2;

a(N)=(-x(N-3)+4\*x(N-2)-5\*x(N-1)+2\*x(N))/dt^2;

for d=2:N-1

a(d)=(x(d+1)-2\*x(d)+x(d-1))/dt^2;

end

end

**-To motion:**

clc

clear variables

close all

vid= VideoReader('FowlerFlapC.mp4');

nFrames= vid.NumberOfFrames;

A= zeros(nFrames,2);

red= 0;

green= 13;

blue= 18;

range= 10;

%%%%%%%%%%%%%%%%%%%%%%%%%%

for k=1:nFrames

a= read(vid,k);

fr= a(:,:,1);

fg= a(:,:,2);

fb= a(:,:,3);

P=((red+range>fr)&(fr>red-range)&(green+range>fg)&(fg>green-range)&(blue+range>fb)&(fb>blue-range));

[y1,x1]=find(P==1);

Px=mean(x1);

Py=mean(y1);

A(k,1)=Px;

A(k,2)=Py;

imshow(a);

hold on

title(['Step',num2str(k),':X = ',num2str(Px),': Y = ',num2str(Py)]);

plot(Px,Py,'ro','linewidth',2)

pause(0.1)

end

A\_mm=A\*0.34;

figure();hold on

title('Raw tracking data');

xlabel('X [mm]');

ylabel('Y [mm]');

plot(A\_mm(:,1),A\_mm(:,2.));

xf=smooth(A\_mm(:,1));

yf=smooth(A\_mm(:,2));

figure();hold on

**-To Find positions and the other equations and their plots:**

%Eray Sozer, Eren Berke Demirbas, Batuhan Yalcin

%Mech 206 Project

clear all

close all

clc

tol = 0.000000001;

max\_it = 1000;

b=313.4481/125;

%Distances

rEA1=39.0448/b;

rA1A2=25.115/b;

rA2D=33.542/b;

rED=29.975/b;

rEE1=35.217/b;

rEE2=78.698/b;

rE1B=77.745/b;

rLB=114.220/b;

rBB1=41.688/b;

rB1J=64.215/b;

rE2J=105.907/b;

rB1F=52.901/b;

rJJ1=65.1197/b;

rJ1G=10.913/b;

rFG=59.075/b;

rE2J1= 161.7648/b;

rJ1H=36.605/b;

rGH=46.638/b;

rPH=57.408/b;

rAGA1=9.3439/b;

rBGL=79.0985/b;

rCGE1=38.8722/b;

rDGA2=16.7150/b;

rEGE=34.0143/b;

rFGB1=34.0637/b;

rGGF=29.5373/b;

rHGJ1=12.6874/b;

rJGE2=124.209/b;

%Mass Moment of Inertias Masses

IA=26.72237/b^4; mA=0.2631/b^2;

IB=3291.50/b^4; mB=1.7138/b^2;

IC=269.4569/b^4; mC=0.4967/b^2;

ID=27.5096/b^4; mD=0.2170/b^2;

IE=428.607/b^4; mE=0.8102/b^2;

IF=208.553/b^4; mF=0.4934/b^2;

IG=125.284/b^4; mG=0.3602/b^2;

IH=56.8824/b^4; mH=0.2685/b^2;

II=77.5299/b^4; mI=0.2539/b^2;

% IJ=6482; mJ=;

TOTAL\_MASS=mA+mB+mC+mD+mE+mF+mG+mH+mI;

thetadot = (5.4836+90-2.0307)/360\*2\*pi/9; %% angular speed (rad/s)

period = 2\*pi/thetadot; %% time to take for a cycle of driving arm

n = 360; %% number of intervals in a cycle

cycle = (5.4836+90-2.0307)/360\*2\*pi; %% number of cycles to animate

N = n+1;

t = [0:cycle/thetadot/n:cycle/thetadot]; % t = time

dt=t(2)-t(1);

A1x = 0;

A1y = 0;

Ex = -26.9/b;

Ey = 28.3/b;

Lx =3.3634/b;

Ly =-44.0812/b;

theta0 = (180-5.4836)/180\*pi;

THETA = thetadot\*t + theta0;

THETA = mod(THETA,2\*pi);

ALPHA = zeros(1,N);

BETA = zeros(1,N);

angles = [195/180\*pi, 160/180\*pi]'; % Initial guess for alpha and beta

for i = 1:1:N

A2x = rA1A2\*cos(THETA(i));

A2y = rA1A2\*sin(THETA(i));

iter = 1;

while (iter <= max\_it)

alpha = angles(1);

beta = angles(2);

f = [ (Ex+rED\*cos(alpha))-(A2x+rA2D\*cos(beta))

(Ey+rED\*sin(alpha))-(A2y+rA2D\*sin(beta))];

J = [ (-rED\*sin(alpha)) ( rA2D\*sin(beta))

( rED\*cos(alpha)) (-rA2D\*cos(beta)) ];

angles\_new = angles -J\f;

err = norm(angles\_new - angles);

if err <= tol

angles\_next = angles\_new;

break

else

angles = angles\_new;

end

iter = iter + 1;

end

if (iter > max\_it), error('Newton method did not converge'); end

angles = mod(angles\_next,2\*pi);

ALPHA(i) = angles(1);

BETA(i) = angles(2);

end

A1X = zeros(size(t));

A1Y = zeros(size(t));

A2X = rA1A2\*cos(THETA);

A2Y = rA1A2\*sin(THETA);

EX = Ex\*ones(size(t));

EY = Ey\*ones(size(t));

DX1 = A2X + rA2D\*cos(BETA);

DY1 = A2Y + rA2D\*sin(BETA);

DX2 = EX + rED\*cos(ALPHA);

DY2 = EY + rED\*sin(ALPHA);

E1X=EX+rEE1\*cos(ALPHA-22.704/180\*pi);

E1Y=EY+rEE1\*sin(ALPHA-22.704/180\*pi);

E2X=EX+rEE2\*cos(ALPHA-(22.704-22.321)/180\*pi);

E2Y=EY+rEE2\*sin(ALPHA-(22.704-22.321)/180\*pi);

PSI = zeros(1,N);

GAMMA = zeros(1,N);

angles = [175/180\*pi, 230/180\*pi]'; % Initial guess for psi and gamma

for i = 1:1:N

E1x = E1X(i);

E1y = E1Y(i);

iter = 1;

while (iter <= max\_it)

psi = angles(1);

gamma = angles(2);

f = [ (E1x + rE1B\*cos(gamma)) - (rLB\*cos(psi) + Lx);

(E1y + rE1B\*sin(gamma)) - (rLB\*sin(psi) + Ly)];

J = [ ( rLB\*sin(psi)) (-rE1B\*sin(gamma))

(-rLB\*cos(psi)) ( rE1B\*cos(gamma)) ];

angles\_new = angles -J\f;

err = norm(angles\_new - angles);

if err <= tol

angles\_next = angles\_new;

break

else

angles = angles\_new;

end

iter = iter + 1;

end

if (iter > max\_it), error('Newton method did not converge'); end

angles = mod(angles\_next,2\*pi);

PSI(i) = angles(1);

GAMMA(i) = angles(2);

end

LX=Lx\*ones(size(t));

LY=Ly\*ones(size(t));

BX1=E1X + rE1B\*cos(GAMMA);

BY1=E1Y + rE1B\*sin(GAMMA);

BX2=LX + rLB\*cos(PSI);

BY2=LY + rLB\*sin(PSI);

B1X=BX1 + rBB1\*cos(PSI+10.1/180\*pi);

B1Y=BY1 + rBB1\*sin(PSI+10.1/180\*pi);

KAPPA = zeros(1,N);

MU = zeros(1,N);

angles = [140/180\*pi, 180.6492/180\*pi]';

for i = 1:1:N

B1x=B1X(i);

B1y=B1Y(i);

E2x=E2X(i);

E2y=E2Y(i);

iter = 1;

while (iter <= max\_it)

kappa = angles(1);

mu = angles(2);

f = [ (B1x+rB1J\*cos(kappa))-(E2x+rE2J\*cos(mu));

(B1y+rB1J\*sin(kappa))-(E2y+rE2J\*sin(mu));];

J = [ ( -rB1J\*sin(kappa)) (rE2J\*sin(mu))

(rB1J\*cos(kappa)) (-rE2J\*cos(mu)) ];

angles\_new = angles -J\f;

err = norm(angles\_new - angles);

if err <= tol

angles\_next = angles\_new;

break

else

angles = angles\_new;

end

iter = iter + 1;

end

if (iter > max\_it), error('Newton method did not converge'); end

angles = mod(angles\_next,2\*pi);

KAPPA(i) = angles(1);

MU(i) = angles(2);

end

JX1=B1X + rB1J\*cos(KAPPA);

JY1=B1Y + rB1J\*sin(KAPPA);

JX2=E2X + rE2J\*cos(MU);

JY2=E2Y + rE2J\*sin(MU);

FX=B1X + rB1F\*cos(KAPPA+1.947/180\*pi);

FY=B1Y + rB1F\*sin(KAPPA+1.947/180\*pi);

J1X=E2X + rE2J1\*cos(MU+14.6929/180\*pi);

J1Y=E2Y + rE2J1\*sin(MU+14.6929/180\*pi);

EPSILON = zeros(1,N);

OMRICON = zeros(1,N);

angles = [205/180\*pi, 80/180\*pi]';

for i = 1:1:N

Fx=FX(i);

Fy=FY(i);

J1x=J1X(i);

J1y=J1Y(i);

iter = 1;

while (iter <= max\_it)

epsilon = angles(1);

omricon = angles(2);

f = [ (J1x + rJ1G\*cos(omricon))-(Fx + rFG\*cos(epsilon));

(J1y + rJ1G\*sin(omricon))-(Fy + rFG\*sin(epsilon))];

J = [ (rFG\*sin(epsilon)) (-rJ1G\*sin(omricon))

(-rFG\*cos(epsilon)) (rJ1G\*cos(omricon))];

angles\_new = angles -J\f;

err = norm(angles\_new - angles);

if err <= tol

angles\_next = angles\_new;

break

else

angles = angles\_new;

end

iter = iter + 1;

end

if (iter > max\_it), error('Newton method did not converge'); end

angles = mod(angles\_next,2\*pi);

EPSILON(i) = angles(1);

OMRICON(i) = angles(2);

end

GX1=J1X + rJ1G\*cos(OMRICON);

GY1=J1Y + rJ1G\*sin(OMRICON);

GX2=FX + rFG\*cos(EPSILON);

GY2=FY + rFG\*sin(EPSILON);

HX=GX1 + rGH\*cos(OMRICON+pi+20.373/180\*pi);

HY=GY1 + rGH\*sin(OMRICON+pi+20.373/180\*pi);

%PATH POINTS

PATH1X= E2X + 231.0308\*cos(MU+ 22.0076/180\*pi)/b;

PATH2X= E2X + 227.8916\*cos(MU+ 21.4820/180\*pi)/b;

PATH3X= E2X + 226.3214\*cos(MU+ 21.3091/180\*pi)/b;

PATH4X= E2X + 224.2858\*cos(MU+ 21.0276/180\*pi)/b;

PATH5X= E2X + 219.2622\*cos(MU+ 20.3070/180\*pi)/b;

PATH6X= E2X + 217.3498\*cos(MU+ 20.0928/180\*pi)/b;

PATH7X= E2X + 213.5835\*cos(MU+ 19.5721/180\*pi)/b;

PATH8X= E2X + 209.9536\*cos(MU+ 19.1397/180\*pi)/b;

PATH9X= E2X + 205.6941\*cos(MU+ 18.7227/180\*pi)/b;

PATH10X= E2X + 202.8184\*cos(MU+ 18.4349/180\*pi)/b;

PATH11X= E2X + 198.5191\*cos(MU+ 18.1279/180\*pi)/b;

PATH12X= E2X + 193.4287\*cos(MU+ 17.7468/180\*pi)/b;

PATH13X= E2X + 190.5485\*cos(MU+ 17.4764/180\*pi)/b;

PATH14X= E2X + 188.0263\*cos(MU+ 17.3256/180\*pi)/b;

PATH1Y= E2Y + 231.0308\*sin(MU+ 22.0076/180\*pi)/b;

PATH2Y= E2Y + 227.8916\*sin(MU+ 21.4820/180\*pi)/b;

PATH3Y= E2Y + 226.3214\*sin(MU+ 21.3091/180\*pi)/b;

PATH4Y= E2Y + 224.2858\*sin(MU+ 21.0276/180\*pi)/b;

PATH5Y= E2Y + 219.2622\*sin(MU+ 20.3070/180\*pi)/b;

PATH6Y= E2Y + 217.3498\*sin(MU+ 20.0928/180\*pi)/b;

PATH7Y= E2Y + 213.5835\*sin(MU+ 19.5721/180\*pi)/b;

PATH8Y= E2Y + 209.9536\*sin(MU+ 19.1397/180\*pi)/b;

PATH9Y= E2Y + 205.6941\*sin(MU+ 18.7227/180\*pi)/b;

PATH10Y= E2Y + 202.8184\*sin(MU+ 18.4349/180\*pi)/b;

PATH11Y= E2Y + 198.5191\*sin(MU+ 18.1279/180\*pi)/b;

PATH12Y= E2Y + 193.4287\*sin(MU+ 17.7468/180\*pi)/b;

PATH13Y= E2Y + 190.5485\*sin(MU+ 17.4764/180\*pi)/b;

PATH14Y= E2Y + 188.0263\*sin(MU+ 17.3256/180\*pi)/b;

PATHX=[PATH1X; PATH2X; PATH3X; PATH4X; PATH5X; PATH6X; PATH7X; PATH8X; PATH9X; PATH10X; PATH11X; PATH12X; PATH13X; PATH14X]';

PATHY=[PATH1Y; PATH2Y; PATH3Y; PATH4Y; PATH5Y; PATH6Y; PATH7Y; PATH8Y; PATH9Y; PATH10Y; PATH11Y; PATH12Y; PATH13Y; PATH14Y]';

for i=1:length(J1Y)

xx=linspace(PATH1X(i),PATH12X(i),100001);

yy=spline(PATHX(i,:),PATHY(i,:),xx);

Distance=sqrt((xx-HX(i)).^2 + (yy-HY(i)).^2);

a=abs(Distance-rPH);

PX(i)=xx(min(a)==a);

PY(i)=yy(min(a)==a);

end

TAU=acos((PX-HX)./sqrt((PX-HX).^2+(PY-HY).^2));

%GRAVITY CENTERS

AGX=A1X + rAGA1\*cos(THETA); AGY=A1Y + rAGA1\*sin(THETA);

BGX=LX + rBGL\*cos(PSI-2.3697/180\*pi); BGY=LY + rBGL\*sin(PSI-2.3697/180\*pi);

CGX=E1X + rCGE1\*cos(GAMMA); CGY=E1Y + rCGE1\*sin(GAMMA);

DGX=A2X + rDGA2\*cos(BETA+3.0615/180\*pi); DGY=A2Y + rDGA2\*sin(BETA+3.0615/180\*pi);

EGX=EX + rEGE\*cos(ALPHA-4.2256/180\*pi); EGY=EY + rEGE\*sin(ALPHA-4.2256/180\*pi);

FGX=B1X + rFGB1\*cos(KAPPA-0.1075/180\*pi); FGY=B1Y + rFGB1\*sin(KAPPA-0.1075/180\*pi);

GGX=FX + rGGF\*cos(EPSILON); GGY=FY + rGGF\*sin(EPSILON);

HGX=J1X + rHGJ1\*cos(OMRICON-141.5307/180\*pi); HGY=J1Y + rHGJ1\*sin(OMRICON-141.5307/180\*pi);

IGX = HX + (PX-HX)/2; IGY = HY + (PY-HY)/2;

JGX=E2X + rJGE2\*cos(MU+16.532/180\*pi); JGY=E2Y + rJGE2\*sin(MU+16.532/180\*pi);

% VELOCITY AND ACCELERATION OF JOINTS

load('eren\_acceleration')

a=1;

VDX=first\_derivative(DX1,dt);

VDY=first\_derivative(DY1,dt);

ADX=second\_derivative(DX1,dt);

ADY=second\_derivative(DY1,dt);

VDJ=sqrt(VDX.^2+VDY.^2);

ADJ=sqrt(ADX.^2+ADY.^2);

figure(a)

subplot(3,1,1)

plot(DX1,DY1), title('Position of D Joint'), xlabel('x(cm)'), ylabel('y(cm)'), grid on

subplot(3,1,2)

plot(t,VDJ,velocityD(:,3)',velocityD(:,4)'+mean(-velocityD(:,4)+VDJ)), title('Velocity of D Joint'), ylabel('V(cm/s)'), xlabel('t(s)'), grid on

subplot(3,1,3)

plot(t,ADJ,accelerationD(:,3)',accelerationD(:,4)'+mean(-accelerationD(:,4)+ADJ)), title('Acceleration of D Joint'), ylabel('A(cm/s^2)'), xlabel('t(s)'), grid on

a=a+1;

VB1X=first\_derivative(B1X,dt);

VB1Y=first\_derivative(B1Y,dt);

AB1X=second\_derivative(B1X,dt);

AB1Y=second\_derivative(B1Y,dt);

VB1J=sqrt(VB1X.^2+VB1Y.^2);

AB1J=sqrt(AB1X.^2+AB1Y.^2);

figure(a)

subplot(3,1,1)

plot(B1X,B1Y), title('Position of B1 Joint'), xlabel('x(cm)'), ylabel('y(cm)'), grid on

subplot(3,1,2)

plot(t,VB1J,velocityB1(:,3)',velocityB1(:,4)'+mean(-velocityB1(:,4)+VB1J)), title('Velocity of B1 Joint'), ylabel('V(cm/s)'), xlabel('t(s)'), grid on

subplot(3,1,3)

plot(t,AB1J,accelerationB1(:,3)',accelerationB1(:,4)'+mean(-accelerationB1(:,4)+AB1J)), title('Acceleration of B1 Joint'), ylabel('A(cm/s^2)'), xlabel('t(s)'), grid on

a=a+1;

VE2X=first\_derivative(E2X,dt);

VE2Y=first\_derivative(E2Y,dt);

AE2X=second\_derivative(E2X,dt);

AE2Y=second\_derivative(E2Y,dt);

VE2J=sqrt(VE2X.^2+VE2Y.^2);

AE2J=sqrt(AE2X.^2+AE2Y.^2);

figure(a)

subplot(3,1,1)

plot(E2X,E2Y), title('Position of E2 Joint'), xlabel('x(cm)'), ylabel('y(cm)'), grid on

subplot(3,1,2)

plot(t,VE2J,velocityE2(:,3)',velocityE2(:,4)'+mean(-velocityE2(:,4)+VE2J)), title('Velocity of E2 Joint'), ylabel('V(cm/s)'), xlabel('t(s)'), grid on

subplot(3,1,3)

plot(t,AE2J,accelerationE2(:,3)',accelerationE2(:,4)'+mean(-accelerationE2(:,4)+AE2J)), title('Acceleration of E2 Joint'), ylabel('A(cm/s^2)'), xlabel('t(s)'), grid on

a=a+1;

VJX=first\_derivative(JX1,dt);

VJY=first\_derivative(JY1,dt);

AJX=second\_derivative(JX1,dt);

AJY=second\_derivative(JY1,dt);

VJJ=sqrt(VJX.^2+VJY.^2);

AJJ=sqrt(AJX.^2+AJY.^2);

figure(a)

subplot(3,1,1)

plot(JX1,JY1), title('Position of J Joint'), xlabel('x(cm)'), ylabel('y(cm)'), grid on

subplot(3,1,2)

plot(t,VJJ,velocityJ(:,3)',velocityJ(:,4)'+mean(-velocityJ(:,4)+VJJ)), title('Velocity of J Joint'), ylabel('V(cm/s)'), xlabel('t(s)'), grid on

subplot(3,1,3)

plot(t,AJJ,accelerationJ(:,3)',accelerationJ(:,4)'+mean(-accelerationJ(:,4)+AJJ)), title('Acceleration of J Joint'), ylabel('A(cm/s^2)'), xlabel('t(s)'), grid on

a=a+1;

VHX=first\_derivative(HX,dt);

VHY=first\_derivative(HY,dt);

AHX=second\_derivative(HX,dt);

AHY=second\_derivative(HY,dt);

VHJ=sqrt(VHX.^2+VHY.^2);

AHJ=sqrt(AHX.^2+AHY.^2);

figure(a)

subplot(3,1,1)

plot(HX,HY), title('Position of H Joint'), xlabel('x(cm)'), ylabel('y(cm)'), grid on

subplot(3,1,2)

plot(t,VHJ,velocityH(:,3)',velocityH(:,4)'+mean(-velocityH(:,4)+VHJ)), title('Velocity of H Joint'), ylabel('V(cm/s)'), xlabel('t(s)'), grid on, legend('Matlab Calculation','Motion Track')

subplot(3,1,3)

plot(t,AHJ,accelerationH(:,3)',accelerationH(:,4)'+mean(-accelerationH(:,4)+AHJ)), title('Acceleration of H Joint'), ylabel('A(cm/s^2)'), xlabel('t(s)'), grid on

a=a+1;

%MOTION TRACK

factor=33.542/81.3749;

MT\_A2=factor\*load('kirmizi.dat')/b; MT\_A2X=MT\_A2(:,1)'-336.9747+202.5925; MT\_A2Y=-(MT\_A2(:,2)'-69.7681)-40.7848;

MT\_D=factor\*load('mavi.dat')/b; MT\_DX=MT\_D(:,1)'-336.9747+202.5925; MT\_DY=-(MT\_D(:,2)'-69.7681)-40.7848;

MT\_E1=factor\*load('pembe.dat')/b; MT\_E1X=MT\_E1(:,1)'-336.9747+202.5925; MT\_E1Y=-(MT\_E1(:,2)'-69.7681)-40.7848;

MT\_B=factor\*load('siyah.dat')/b; MT\_BX=MT\_B(:,1)'-336.9747+202.5925; MT\_BY=-(MT\_B(:,2)'-69.7681)-40.7848;

MT\_E2=factor\*load('cyan.dat')/b; MT\_E2X=MT\_E2(:,1)'-336.9747+202.5925; MT\_E2Y=-(MT\_E2(:,2)'-69.7681)-40.7848;

MT\_B1=factor\*load('yesil.dat')/b; MT\_B1X=MT\_B1(:,1)'-336.9747+202.5925; MT\_B1Y=-(MT\_B1(:,2)'-69.7681)-40.7848;

MT\_J=factor\*load('sari.dat')/b; MT\_JX=MT\_J(:,1)'-336.9747+202.5925; MT\_JY=-(MT\_J(:,2)'-69.7681)-40.7848;

MT\_F=factor\*load('turuncu.dat')/b; MT\_FX=MT\_F(:,1)'-336.9747+202.5925; MT\_FY=-(MT\_F(:,2)'-69.7681)-40.7848;

MT\_J1=factor\*load('beyaz.dat')/b; MT\_J1X=MT\_J1(:,1)'-336.9747+202.5925; MT\_J1Y=-(MT\_J1(:,2)'-69.7681)-40.7848;

MT\_H=factor\*load('kahve.dat')/b; MT\_HX=MT\_H(:,1)'-336.9747+202.5925; MT\_HY=-(MT\_H(:,2)'-69.7681)-40.7848;

MT\_G=factor\*load('mor.dat')/b; MT\_GX=MT\_G(:,1)'-336.9747+202.5925; MT\_GY=-(MT\_G(:,2)'-69.7681)-40.7848;

MT\_P=load('flap.dat')/b; MT\_PX=MT\_P(:,1)'-(MT\_P(1,1)-PX(1)); MT\_PY=-(MT\_P(:,2)'-(MT\_H(1,2)-HY(1)));

pfactor=(PX(end)-PX(1))/(MT\_PX(end)-MT\_PX(1)); MT\_PX=pfactor\*MT\_PX-77.94; MT\_PY=pfactor\*MT\_PY-12.58;

%FORCES

FA=mA\*sqrt(second\_derivative(AGX,dt).^2+second\_derivative(AGY,dt).^2); FA=mean(FA)\*ones(size(FA));

FB=mB\*sqrt(second\_derivative(BGX,dt).^2+second\_derivative(BGY,dt).^2);

FC=mC\*sqrt(second\_derivative(CGX,dt).^2+second\_derivative(CGY,dt).^2);

FD=mD\*sqrt(second\_derivative(DGX,dt).^2+second\_derivative(DGY,dt).^2);

FE=mE\*sqrt(second\_derivative(EGX,dt).^2+second\_derivative(EGY,dt).^2);

FF=mF\*sqrt(second\_derivative(FGX,dt).^2+second\_derivative(FGY,dt).^2);

FG=mG\*sqrt(second\_derivative(GGX,dt).^2+second\_derivative(GGY,dt).^2);

FH=mH\*sqrt(second\_derivative(HGX,dt).^2+second\_derivative(HGY,dt).^2);

FI=mI\*sqrt(second\_derivative(IGX,dt).^2+second\_derivative(IGY,dt).^2);

% FJ=mJ\*sqrt(second\_derivative(JGX,dt).^2+second\_derivative(JGY,dt).^2);

TOTAL\_FORCE=FA+FB+FC+FD+FE+FF+FG+FH+FI;

%MOMENTS

MA=abs(IA\*second\_derivative(THETA,dt)); MA=zeros(size(MA));

MB=abs(IB\*second\_derivative(PSI,dt));

MC=abs(IC\*second\_derivative(GAMMA,dt));

MD=abs(ID\*second\_derivative(BETA,dt));

ME=abs(IE\*second\_derivative(ALPHA,dt));

MF=abs(IF\*second\_derivative(KAPPA,dt));

MG=abs(IG\*second\_derivative(EPSILON,dt));

MH=abs(IH\*second\_derivative(OMRICON,dt));

MI=abs(IH\*second\_derivative(TAU,dt));

% MJ=abs(IJ\*second\_derivative(MU,dt));

TOTAL\_MOMENT=MA+MB+MC+MD+ME+MF+MG+MH+MI;

%POWERS

PA= abs(FA .\* sqrt(first\_derivative(AGX,dt).^2 + first\_derivative(AGY,dt).^2)) + abs(MA .\* first\_derivative(THETA,dt)); PA=mean(PA)\*ones(size(PA));

PB= abs(FB .\* sqrt(first\_derivative(BGX,dt).^2 + first\_derivative(BGY,dt).^2)) + abs(MB .\* first\_derivative(PSI,dt));

PC= abs(FC .\* sqrt(first\_derivative(CGX,dt).^2 + first\_derivative(CGY,dt).^2)) + abs(MC .\* first\_derivative(GAMMA,dt));

PD= abs(FD .\* sqrt(first\_derivative(DGX,dt).^2 + first\_derivative(DGY,dt).^2)) + abs(MD .\* first\_derivative(BETA,dt));

PE= abs(FE .\* sqrt(first\_derivative(EGX,dt).^2 + first\_derivative(EGY,dt).^2)) + abs(ME .\* first\_derivative(ALPHA,dt));

PF= abs(FF .\* sqrt(first\_derivative(FGX,dt).^2 + first\_derivative(FGY,dt).^2)) + abs(MF .\* first\_derivative(KAPPA,dt));

PG= abs(FG .\* sqrt(first\_derivative(GGX,dt).^2 + first\_derivative(GGY,dt).^2)) + abs(MG .\* first\_derivative(EPSILON,dt));

PH= abs(FH .\* sqrt(first\_derivative(HGX,dt).^2 + first\_derivative(HGY,dt).^2)) + abs(MH .\* first\_derivative(OMRICON,dt));

PI= abs(FI .\* sqrt(first\_derivative(IGX,dt).^2 + first\_derivative(IGY,dt).^2)) + abs(MI .\* first\_derivative(TAU,dt));

%PJ= abs(FJ .\* sqrt(first\_derivative(JGX,dt).^2 + first\_derivative(JGY,dt)^2)) + abs(MJ .\* first\_derivative(THETA,dt));

TOTAL\_POWER=PA+PB+PC+PD+PE+PF+PG+PH+PI;

%ENERGY

UA=0.5\*mA\*(first\_derivative(AGX,dt).^2 + first\_derivative(AGY,dt).^2); UA=mean(UA)\*ones(size(UA));

UE=0.5\*mE\*(first\_derivative(EGX,dt).^2 + first\_derivative(EGY,dt).^2);

UF=0.5\*mF\*(first\_derivative(FGX,dt).^2 + first\_derivative(FGY,dt).^2);

UH=0.5\*mH\*(first\_derivative(HGX,dt).^2 + first\_derivative(HGY,dt).^2);

% figure(a)

% subplot(3,1,1)

% plot(t,TOTAL\_FORCE), title('Total Force'), xlabel('F(kg\*cm/s^2)'), ylabel('t(s)'), grid on

% subplot(3,1,2)

% plot(t,TOTAL\_MOMENT), title('Total Moment'), ylabel('V(cm/s)'), xlabel('t(s)'), grid on

% subplot(3,1,3)

% plot(t,TOTAL\_POWER), title('Total Power'), ylabel('P(kg\*cm^2/s^2)'), xlabel('t(s)'), grid on

% a=a+1;

figure(a)

subplot(2,2,1)

plot(t,FA), title('Force: Link A'), xlabel('t(s)'), ylabel('F(kg\*cm/s^2)'), grid on

subplot(2,2,2)

plot(t,MA), title('Moment: Link A'), xlabel('t(s)'), ylabel('M(kg\*cm/s^2)'), grid on

subplot(2,2,3)

plot(t,UA), title('Energy: Link A'), xlabel('t(s)'), ylabel('U(kg\*cm^2/s^2)'), grid on

subplot(2,2,4)

plot(t,PA), title('Power: Link A'), xlabel('t(s)'), ylabel('P(kg\*cm^2/s^3)'), grid on

a=a+1;

figure(a)

subplot(2,2,1)

plot(t,FE), title('Force: Link E'), xlabel('t(s)'), ylabel('F(kg\*cm/s^2)'), grid on

subplot(2,2,2)

plot(t,ME), title('Moment: Link E'), xlabel('t(s)'), ylabel('M(kg\*cm/s^2)'), grid on

subplot(2,2,3)

plot(t,UE), title('Energy: Link E'), xlabel('t(s)'), ylabel('U(kg\*cm^2/s^2)'), grid on

subplot(2,2,4)

plot(t,PE), title('Power: Link E'), xlabel('t(s)'), ylabel('P(kg\*cm^2/s^3)'), grid on

a=a+1;

figure(a)

subplot(2,2,1)

plot(t,FF), title('Force: Link F'), xlabel('t(s)'), ylabel('F(kg\*cm/s^2)'), grid on

subplot(2,2,2)

plot(t,MF), title('Moment: Link F'), xlabel('t(s)'), ylabel('M(kg\*cm/s^2)'), grid on

subplot(2,2,3)

plot(t,UF), title('Energy: Link F'), xlabel('t(s)'), ylabel('U(kg\*cm^2/s^2)'), grid on

subplot(2,2,4)

plot(t,PF), title('Power: Link F'), xlabel('t(s)'), ylabel('P(kg\*cm^2/s^3)'), grid on

a=a+1;

figure(a)

subplot(2,2,1)

plot(t,FH), title('Force: Link H'), xlabel('t(s)'), ylabel('F(kg\*cm/s^2)'), grid on

subplot(2,2,2)

plot(t,MH), title('Moment: Link H'), xlabel('t(s)'), ylabel('M(kg\*cm/s^2)'), grid on

subplot(2,2,3)

plot(t,UH), title('Energy: Link H'), xlabel('t(s)'), ylabel('U(kg\*cm^2/s^2)'), grid on

subplot(2,2,4)

plot(t,PH), title('Power: Link H'), xlabel('t(s)'), ylabel('P(kg\*cm^2/s^3)'), grid on

a=a+1;

figure(a)

subplot(3,4,1)

plot(MT\_A2X,MT\_A2Y,A2X,A2Y), title('A2'), grid on,

subplot(3,4,2)

plot(MT\_DX,MT\_DY,DX1,DY1), title('D'), grid on

subplot(3,4,3)

plot(MT\_E1X,MT\_E1Y,E1X,E1Y), title('E'), grid on

subplot(3,4,4)

plot(MT\_BX,MT\_BY,BX1,BY1), title('B'), grid on

subplot(3,4,5)

plot(MT\_E2X,MT\_E2Y,E2X,E2Y), title('E2'), grid on

subplot(3,4,6)

plot(MT\_B1X,MT\_B1Y,B1X,B1Y), title('B1'), grid on

subplot(3,4,7)

plot(MT\_JX,MT\_JY,JX1,JY1), title('J'), grid on

subplot(3,4,8)

plot(MT\_FX,MT\_FY,FX,FY), title('F'), grid on

subplot(3,4,9)

plot(MT\_J1X,MT\_J1Y,J1X,J1Y), title('J1'), grid on

subplot(3,4,10)

plot(MT\_HX,MT\_HY,HX,HY), title('H'), grid on

subplot(3,4,11)

plot(MT\_GX,MT\_GY,GX1,GY1), title('G'), grid on

subplot(3,4,12)

plot(MT\_PX,MT\_PY,PX,PY), title('P'), grid on

a=a+1;

% figure(a)

%

% pause(2)

% for i=1:3:N

% text\_plot = ['time = ',num2str(t(i),'%7.4f'),' s'];

% plot([A1X(i) A2X(i)],[A1Y(i) A2Y(i)],'r-', [A2X(i) DX1(i)],[A2Y(i) DY1(i)],'b-', [EX(i) DX1(i)],[EY(i) DY1(i)],'m-',[EX(i) E1X(i)],[EY(i) E1Y(i)],'m-',[E1X(i) E2X(i)],[E1Y(i) E2Y(i)],'m-',[E2X(i) DX1(i)],[E2Y(i) DY1(i)],'m-',[E1X(i) BX1(i)],[E1Y(i) BY1(i)],'g-',[LX(i) BX1(i)],[LY(i) BY1(i)],'y-',[B1X(i) BX1(i)],[B1Y(i) BY1(i)],'y-',[B1X(i) FX(i)],[B1Y(i) FY(i)],'r-',[JX1(i) FX(i)],[JY1(i) FY(i)],'r-',[E2X(i) JX1(i)],[E2Y(i) JY1(i)],'b-',[FX(i) GX1(i)],[FY(i) GY1(i)],'k-',[J1X(i) GX1(i)],[J1Y(i) GY1(i)],'m-',[J1X(i) HX(i)],[J1Y(i) HY(i)],'m-',[PX(i) HX(i)],[PY(i) HY(i)],'k-',[AGX(i) BGX(i) CGX(i) DGX(i) EGX(i) FGX(i) GGX(i) HGX(i) IGX(i)],[AGY(i) BGY(i) CGY(i) DGY(i) EGY(i) FGY(i) GGY(i) HGY(i) IGY(i)],'ro','linewidth',5)

% grid on, xlabel('x(cm)'), ylabel('y(cm)'), title(text\_plot)

% axis ([-340/b 20/b -218.5/2/b 218.5/2/b])

% drawnow

% pause(0.01)

% end

% a=a+1;