Table of Contents


```
clear;
clc;
```

Initial Position calculations

```
A=[1.4 .485 0];
B=[1.67 0.99 0];
C=[.255  1.035 0];
D=[.285 .055 0];
E=[.195 2.54 0];
F=[-.98 2.57 0];
G=[.05 .2 0];

initalPosition = linkageAnalysis(B,C,E,F);
```

Full Rotation Position Calculator

```
%length of each link/ distance between joints
AB=norm(B-A);
BC=norm(C-B);
CD=norm(D-C);
CE=norm(E-C);
DE=norm(E-D);
BE=norm(E-B);
EF=norm(F-E);
FG=norm(G-F);
```

Circle Method

```
%now obtain the new positions
%initial angle
initialAngle_AB=atan2(B(2)-A(2),B(1)-A(1));

%if this is negative, then we should subtract as shown below

if(initialAngle_AB<0)
```

```
angleAB_horizontal=2*pi+initialAngle_AB; %adjusting the angle to be in
 the ccw direction from the hozizontal
else
    angleAB_horizontal = initialAngle_AB;
end

%Iterate through 360 degrees
for theta=0:1:360
%position analysis
%increase by 1 deg
%new position of B
B_new = vpa(A+[AB*cos(angleAB_horizontal+deg2rad(theta))
 AB*sin(angleAB_horizontal+deg2rad(theta)) 0]);
%new position of C
[C_x,C_y]=circcirc(B_new(1),B_new(2),BC,D(1),D(2),CD);
%checking if circles are not intersecting
circIntersect_x = any(isnan(vpa(C_x))); %checking for Not-a-Number
circIntersect_y = any(isnan(vpa(C_y)));

if circIntersect_x==0 && circIntersect_y==0 % if the circles are not
 intersecting
    C_1=[C_x(1) C_y(1) 0]; %adding the two solutions
    C_2=[C_x(2) C_y(2) 0];
    dist1 = norm(C_1-C);
    dist2 = norm(C_2-C);

    if(dist1<dist2) %checking which new C is closer
      C_new=vpa(C_1);
    else
      C_new=vpa(C_2);
    end

    %new position of E

    [E_x,E_y]=circcirc(C_new(1),C_new(2),CE,D(1),D(2),DE);

    %checking if circles are not intersecting

    circIntersect_x = any(isnan(vpa(E_x))); %checking for Not-a-Number
    circIntersect_y = any(isnan(vpa(E_y)));


    if circIntersect_x==0 && circIntersect_y==0 % if the circles are
 not intersecting
         E_1=[E_x(1) E_y(1) 0]; %adding the two solutions
         E_2=[E_x(2) E_y(2) 0];
         dist1 = norm(E_1-E);
         dist2 = norm(E_2-E);

        if(dist1<dist2)
            E_new=vpa(E_1);
        else
            E_new=vpa(E_2);
        end
```

```
    % new position of F

        [F_x,F_y]=circcirc(E_new(1),E_new(2),EF,G(1),G(2),FG);

          %checking if circles are not intersecting

        circIntersect_x = any(isnan(vpa(F_x))); %checking for Not-a-
Number
        circIntersect_y = any(isnan(vpa(F_y)));

        if circIntersect_x==0 && circIntersect_y==0  % if the circles
 are not intersecting
             F_1=[F_x(1) F_y(1) 0]; %adding the two solutions
             F_2=[F_x(2) F_y(2) 0];
             dist1 = norm(F_1-F);
             dist2 = norm(F_2-F);

             if(dist1<dist2)
                  F_new=vpa(F_1);
             else
                  F_new=vpa(F_2);
             end


             %storing values
             newB_x(theta+1)=double(B_new(1));
             newB_y(theta+1)=double(B_new(2));
             newC_x(theta+1)=double(C_new(1));
             newC_y(theta+1)=double(C_new(2));
             newE_x(theta+1)=double(E_new(1));
             newE_y(theta+1)=double(E_new(2));
             newF_x(theta+1)=double(F_new(1));
             newF_y(theta+1)=double(F_new(2));

                 %saving this into an Excel spreadsheet


            positionsMatrix = [B_new C_new E_new F_new];

            if (theta==0)


 dlmwrite('PositionsAndForceDiffPos.xls',positionsMatrix,'delimiter','\t','precisi
            else
            dlmwrite('PositionsAndForceDiffPos.xls',positionsMatrix,'-
append',...
            'delimiter','\t','precision',4);
             end

        else
            figure

            ax1= subplot(2,2,1);
```

```
            plot(newB_x,newB_y);
            title(ax1,'Joint B')
            ax2=  subplot(2,2,2);
            plot(newC_x,newC_y);
            title(ax2,'Joint C')
            ax3=  subplot(2,2,3);
            plot(newE_x,newE_y);
            title(ax3,'Joint E')
            ax4=  subplot(2,2,4);
            plot(newF_x,newF_y);
            title(ax4,'Joint F')
            fprintf('New position cannot be determined at this angle
 from the initial: %d',theta);
            return
        end
    else
     fprintf('New position cannot be determined at this angle  from
 the initial: %d',theta);
    return
    end
else
    fprintf('New position cannot be determined at this angle:
 %d',theta);
    return
end

%Calculate and record updated joint parameters
values = linkageAnalysis(B,C,E,F);
statics(:,theta+1) = values.staticSol.';
angulars(:,theta+1) = values.angSol.';
dynamics(:,theta+1) = values.dynamicSol.';

%Update joint Positions
B=B_new;
C=C_new;
E=E_new;
F=F_new;

end
```

Plot final joint parameters

```
jointPos = [newB_x;newB_y;newC_x;newC_y;newE_x;newE_y;newF_x;newF_y];
plotJoints(jointPos,statics,angulars,dynamics);

function values = linkageAnalysis(JB,JC,JE,JF)
```

Static Analysis

```
%static force,static torque, dynamic force,
%dynamic torque, velocity of joints, angular velocity of links,
 angular accelerations of
%links, accelerations of joints, positions of joints
```

```
% Assume assembly is made out of aluminium
%coordinates of joints
A=[1.4 .485 0];
B=JB;
C=JC;
D=[.285 .055 0];
E=JE;
F=JF;
G=[.05 .2 0];
%coordinates of link's COM
Hab = [((A(1,1) + B(1,1)) / 2) ((A(1,2) + B(1,2)) / 2) 0];
Hbc = [((B(1,1) + C(1,1)) / 2) ((B(1,2) + C(1,2)) / 2) 0];
Hde = [((D(1,1) + E(1,1)) / 2) ((D(1,2) + E(1,2)) / 2) 0];
Hef = [((E(1,1) + F(1,1)) / 2) ((E(1,2) + F(1,2)) / 2) 0];

%position vectors of COM & relative points
pvHab = Hab-A;
pvHbc = Hbc-B;
pvHde = Hde-D;
pvHef = Hef-E;

%length of each link/ distance between joints
AB=norm(B-A);
BC=norm(C-B);
CD=norm(D-C);
DE=norm(E-D);
BE=norm(E-B);
EF=norm(F-E);
FG=norm(G-F);
LF=1.843; % distance between load and joint F

%position vectors
pvAB=B-A;
pvBC=C-B;
pvCD=D-C;
pvDA=A-D;
pvDE=E-D;
pvEF=F-E;
pvFG=G-F;

unit_GF=-pvFG/FG;
pvFL=unit_GF*LF;
pvGL=pvFL-pvFG; %load from G to L
L=pvGL+G; %location of load
Hlg = [((L(1,1) + G(1,1)) / 2) ((L(1,2) + G(1,2)) / 2) 0]; % location
 of COM
pvHlg = Hlg-G; % position of COM from G to Hlg

%without weight of each link considered
syms Ax Ay Bx By Cx Cy Dx Dy Ex Ey Fx Fy Gx Gy inTorque
fA=[Ax Ay 0];
fB=[Bx By 0];
fC=[Cx Cy 0];
fD=[Dx Dy 0];
```

```
fE=[Ex Ey 0];
fF=[Fx Fy 0];
fG=[Gx Gy 0];
Ta=[0 0 inTorque];

%weight of links in Newtons (assuming the link material is Al 6061 T6)
linkDensity = [0 2710 0]; % kg/m^3
linkWidth = [0 0.10 0]; % m
linkThickness = [0 0.05 0]; % m
jointDiameter = [0 0.06 0]; % m
Wab = (linkDensity .* linkWidth .* linkThickness .* AB .* -9.8); % N
Wbc = (linkDensity .* linkWidth .* linkThickness .* BC .* -9.8); % N
Wcd = (linkDensity .* linkWidth .* linkThickness .* CD .* -9.8); % N
Wde = (linkDensity .* linkWidth .* linkThickness .* DE .* -9.8); % N
Wef = (linkDensity .* linkWidth .* linkThickness .* EF .* -9.8); % N
Wfg = (linkDensity .* linkWidth .* linkThickness .* (LF + FG) .*
 -9.8); % N
Wl = [0 -200 0] ; %given weight of load in NEWTONS
```

Static Solution

```
%Link AB/1
%First equation represents sum of forces
%Second Equation represents sum of moments
eqn1=fA-fB+Wab==0;
eqn2=Ta+cross(pvHab,Wab)+cross(pvAB,-fB)==0;
%Link BC
eqn3=fB-fC+Wbc==0;
eqn4=cross(pvBC,-fC)+cross(pvHbc,Wbc)==0;
%Link DEC
eqn5=fC-fD+fE+Wde==0;
eqn6=cross(pvDE,fE)+cross(pvHde,Wde)+cross(-pvCD,fC)==0;
%Link EF
eqn7=-fE+fF+Wef==0;
eqn8=cross(pvEF,fF)+cross(pvHef,Wef)==0;
%Link FG with load L
eqn9=-fF+fG+Wfg+Wl==0;
eqn10=cross(-pvFG,-fF)+cross(pvGL,Wl)+cross(pvHlg,Wfg)==0;

staticsolution =
 (solve([eqn1,eqn2,eqn3,eqn4,eqn5,eqn6,eqn7,eqn8,eqn9,eqn10],
[Ax,Ay,Bx,By,Cx,Cy,Dx,Dy,Ex,Ey,Fx,Fy,Gx,Gy,inTorque]));

noWeightforce_Ax=double(staticsolution.Ax);
noWeightforce_Ay=double(staticsolution.Ay);
noWeightforce_Bx=double(staticsolution.Bx);
noWeightforce_By=double(staticsolution.By);
noWeightforce_Cx=double(staticsolution.Cx);
noWeightforce_Cy=double(staticsolution.Cy);
noWeightforce_Dx=double(staticsolution.Dx);
noWeightforce_Dy=double(staticsolution.Dy);
noWeightforce_Ex=double(staticsolution.Ex);
noWeightforce_Ey=double(staticsolution.Ey);
```

```
noWeightforce_Fx=double(staticsolution.Fx);
noWeightforce_Fy=double(staticsolution.Fy);
noWeightforce_Gx=double(staticsolution.Gx);
noWeightforce_Gy=double(staticsolution.Gy);
noWeighttorque_T=double(staticsolution.inTorque);

staticsolution =
 [noWeightforce_Ax;noWeightforce_Ay;noWeightforce_Bx;noWeightforce_By;noWeightforc

 noWeightforce_Dx;noWeightforce_Dy;noWeightforce_Ex;noWeightforce_Ey;noWeightforce
    noWeightforce_Gx;noWeightforce_Gy;noWeighttorque_T];
```

position analysis

```
omegaAB=[0 0 (7450/7)/3600*2*pi]; % 7450 parts per 7 hours assuming 1
 revolution is 1 part
alphaAB=[0 0 0]; % input link rotating at a constant velocity

syms omegaBCz omegaDEz omegaEFz omegaFGz alphaBCz alphaDEz alphaEFz alphaFGz
omegaBC=[0 0 omegaBCz];
omegaDE=[0 0 omegaDEz];
omegaEF=[0 0 omegaEFz];
omegaFG=[0 0 omegaFGz];
alphaBC=[0 0 alphaBCz];
alphaDE=[0 0 alphaDEz];
alphaEF=[0 0 alphaEFz];
alphaFG=[0 0 alphaFGz];

eqn11=cross(omegaAB,pvAB)+cross(omegaBC,pvBC)+cross(omegaDE,pvCD)==0;
eqn12=cross(alphaAB,pvAB)+cross(omegaAB,cross(omegaAB,pvAB))+cross(alphaBC,pvBC)+c
eqn13=cross(omegaDE,pvDE)+cross(omegaEF,pvEF)+cross(omegaFG,pvFG)==0;
eqn14=cross(alphaDE,pvDE)+cross(omegaDE,cross(omegaDE,pvDE))+cross(alphaEF,pvEF)+c

positionsolution= (solve([eqn11,eqn12,eqn13,eqn14],
[omegaBCz,omegaDEz,omegaEFz,omegaFGz,alphaBCz,alphaDEz,alphaEFz,alphaFGz]));

angvel_BCz=double(positionsolution.omegaBCz);
angvel_DEz=double(positionsolution.omegaDEz);
angvel_EFz=double(positionsolution.omegaEFz);
angvel_FGz=double(positionsolution.omegaFGz);

angacc_BCz=double(positionsolution.alphaBCz);
angacc_DEz=double(positionsolution.alphaDEz);
angacc_EFz=double(positionsolution.alphaEFz);
angacc_FGz=double(positionsolution.alphaFGz);

positionsolution = [angvel_BCz;angvel_DEz;angvel_EFz;angvel_FGz;
                    angacc_BCz;angacc_DEz;angacc_EFz;angacc_FGz;
                    norm(cross(omegaAB,pvAB));norm(cross([0 0
 angvel_DEz],-pvCD));norm(cross([0 0 angvel_DEz],pvDE));norm(cross([0
 0 angvel_FGz],-pvFG));

 norm(cross(alphaAB,pvAB)+cross(omegaAB,cross(omegaAB,pvAB)));norm(cross([0
```

```
 0 angacc_DEz],-pvCD)+cross([0 0 angvel_DEz],cross([0 0
 angvel_DEz],-pvCD)));norm(cross([0 0 angacc_DEz],pvDE)+cross([0
 0 angvel_DEz],cross([0 0 angvel_DEz],pvDE)));norm(cross([0 0
 angacc_FGz],-pvFG)+cross([0 0 angvel_FGz],cross([0 0 angvel_FGz],-
pvFG)))];


%extra acceleration values

angvel_BC=[0 0 angvel_BCz];
angvel_DE=[0 0 angvel_DEz];
angvel_EF=[0 0 angvel_EFz];
angvel_FG=[0 0 angvel_FGz];
angacc_BC=[0 0 angacc_BCz];
angacc_DE=[0 0 angacc_DEz];
angacc_EF=[0 0 angacc_EFz];
angacc_FG=[0 0 angacc_FGz];

accH_AB=cross(alphaAB,pvHab)+cross(omegaAB,cross(omegaAB,pvHab));
accH_BC=cross(angacc_BC,pvHbc)+cross(angvel_BC,cross(angvel_BC,pvHbc));
accH_DE=cross(angacc_DE,pvHde)+cross(angvel_DE,cross(angvel_DE,pvHde));
accH_EF=cross(angacc_EF,pvHef)+cross(angvel_EF,cross(angvel_EF,pvHef));
accH_GL=cross(angacc_FG,pvHlg)+cross(angvel_FG,cross(angvel_FG,pvHlg));
```

Dynamic Analysis

```
JAB_A=1/12*(Wab(2)/-9.8)*(linkWidth(2)^2+AB^2)+(Wab(2)/-9.8)*norm(pvHab)^2;
JBC_B=1/12*(Wbc(2)/-9.8)*(linkWidth(2)^2+BC^2)+(Wbc(2)/-9.8)*norm(pvHbc)^2;
JDE_D=1/12*(Wde(2)/-9.8)*(linkWidth(2)^2+DE^2)+(Wde(2)/-9.8)*norm(pvHde)^2;
JEF_E=1/12*(Wef(2)/-9.8)*(linkWidth(2)^2+EF^2)+(Wef(2)/-9.8)*norm(pvHef)^2;
JLG_G=1/12*(Wfg(2)/-9.8)*(linkWidth(2)^2+(LF
+FG)^2)+(Wfg(2)/-9.8)*norm(pvHlg)^2;

eqn15=fA-fB+Wab==(Wab(2)/-9.8)*accH_AB;
eqn16=Ta+cross(pvHab,Wab)+cross(pvAB,-fB)==JAB_A*alphaAB;
%Link BC
eqn17=fB-fC+Wbc==(Wbc(2)/-9.8)*accH_BC;
eqn18=cross(pvBC,-fC)+cross(pvHbc,Wbc)==JBC_B*angacc_BC;
%Link DEC
eqn19=fC-fD+fE+Wde==(Wde(2)/-9.8)*accH_DE;
eqn20=cross(pvDE,fE)+cross(pvHde,Wde)+cross(-
pvCD,fC)==JDE_D*angacc_DE;
%Link EF
eqn21=-fE+fF+Wef==(Wef(2)/-9.8)*accH_EF;
eqn22=cross(pvEF,fF)+cross(pvHef,Wef)==JEF_E*angacc_EF;
%Link FG with load L
eqn23=-fF+fG+Wfg+Wl==(Wfg(2)/-9.8)*accH_GL;
eqn24=cross(-pvFG,-
fF)+cross(pvGL,Wl)+cross(pvHlg,Wfg)==JLG_G*angacc_FG;

dynamicsolution =
 (solve([eqn15,eqn16,eqn17,eqn18,eqn19,eqn20,eqn21,eqn22,eqn23,eqn24],
[Ax,Ay,Bx,By,Cx,Cy,Dx,Dy,Ex,Ey,Fx,Fy,Gx,Gy,inTorque]));
```

```
dynamicforce_Ax=double(dynamicsolution.Ax);
dynamicforce_Ay=double(dynamicsolution.Ay);
dynamicforce_Bx=double(dynamicsolution.Bx);
dynamicforce_By=double(dynamicsolution.By);
dynamicforce_Cx=double(dynamicsolution.Cx);
dynamicforce_Cy=double(dynamicsolution.Cy);
dynamicforce_Dx=double(dynamicsolution.Dx);
dynamicforce_Dy=double(dynamicsolution.Dy);
dynamicforce_Ex=double(dynamicsolution.Ex);
dynamicforce_Ey=double(dynamicsolution.Ey);
dynamicforce_Fx=double(dynamicsolution.Fx);
dynamicforce_Fy=double(dynamicsolution.Fy);
dynamicforce_Gx=double(dynamicsolution.Gx);
dynamicforce_Gy=double(dynamicsolution.Gy);
dynamictorque_T=double(dynamicsolution.inTorque);

dynamicsolution =
 [dynamicforce_Ax;dynamicforce_Ay;dynamicforce_Bx;dynamicforce_By;dynamicforce_Cx;

 dynamicforce_Dx;dynamicforce_Dy;dynamicforce_Ex;dynamicforce_Ey;dynamicforce_Fx;d
    dynamicforce_Gx;dynamicforce_Gy;dynamictorque_T];

values.staticSol = staticsolution;
values.angSol = positionsolution;
values.dynamicSol = dynamicsolution;

end
```

Function for plotting joint parameters

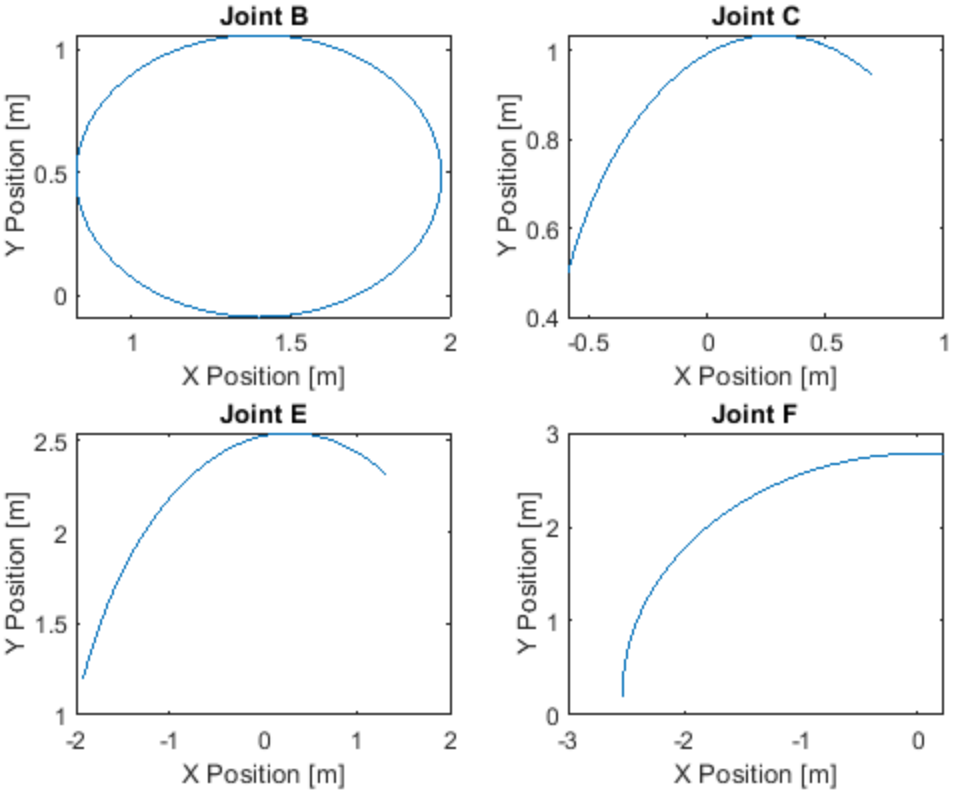
```
function plotJoints(jointPos,statics,angulars,dynamics)
%Plot Joint Positions
theta = 0:1:360;
ax1= subplot(2,2,1);
plot(jointPos(1,:),jointPos(2,:));
title(ax1,'Joint B')
xlabel('X Position [m]')
ylabel('Y Position [m]')
ax2=  subplot(2,2,2);
plot(jointPos(3,:),jointPos(4,:));
title(ax2,'Joint C')
xlabel('X Position [m]')
ylabel('Y Position [m]')
ax3=  subplot(2,2,3);
plot(jointPos(5,:),jointPos(6,:));
title(ax3,'Joint E')
xlabel('X Position [m]')
ylabel('Y Position [m]')
ax4=  subplot(2,2,4);
plot(jointPos(7,:),jointPos(8,:));
title(ax4,'Joint F')
xlabel('X Position [m]')
```

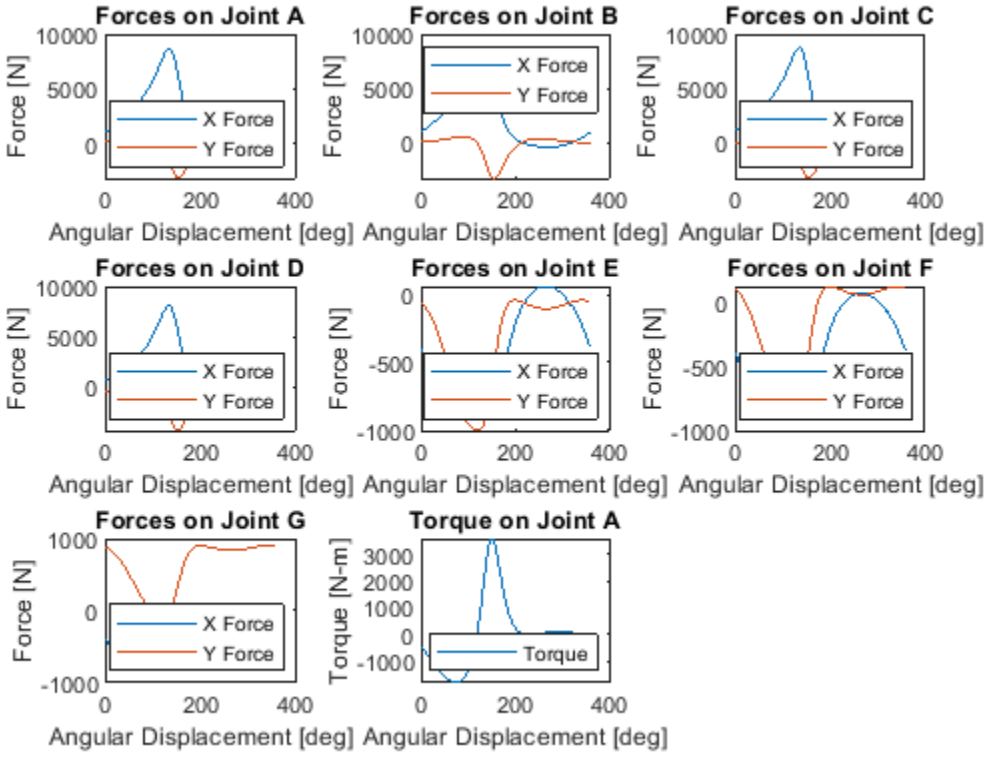
```
ylabel('Y Position [m]')

%Plot Static Forces/Torque
figure('name','Static Joint Forces/Torque');
ax1= subplot(3,3,1);
plot(theta,statics(1,:),theta,statics(2,:));
title(ax1,'Forces on Joint A')
legend('X Force','Y Force','Location','southeast');
xlabel('Angular Displacement [deg]')
ylabel('Force [N]')
ax1= subplot(3,3,2);
plot(theta,statics(3,:),theta,statics(4,:));
title(ax1,'Forces on Joint B')
legend('X Force','Y Force','Location','northeast');
xlabel('Angular Displacement [deg]')
ylabel('Force [N]')
ax1= subplot(3,3,3);
plot(theta,statics(5,:),theta,statics(6,:));
title(ax1,'Forces on Joint C')
legend('X Force','Y Force','Location','southeast');
xlabel('Angular Displacement [deg]')
ylabel('Force [N]')
ax1= subplot(3,3,4);
plot(theta,statics(7,:),theta,statics(8,:));
title(ax1,'Forces on Joint D')
legend('X Force','Y Force','Location','southeast');
xlabel('Angular Displacement [deg]')
ylabel('Force [N]')
ax1= subplot(3,3,5);
plot(theta,statics(9,:),theta,statics(10,:));
title(ax1,'Forces on Joint E')
legend('X Force','Y Force','Location','southeast');
xlabel('Angular Displacement [deg]')
ylabel('Force [N]')
ax1= subplot(3,3,6);
plot(theta,statics(11,:),theta,statics(12,:));
title(ax1,'Forces on Joint F')
legend('X Force','Y Force','Location','southeast');
xlabel('Angular Displacement [deg]')
ylabel('Force [N]')
ax1= subplot(3,3,7);
plot(theta,statics(13,:),theta,statics(14,:));
title(ax1,'Forces on Joint G')
legend('X Force','Y Force','Location','southeast');
xlabel('Angular Displacement [deg]')
ylabel('Force [N]')
ax1= subplot(3,3,8);
plot(theta,statics(15,:));
title(ax1,'Torque on Joint A')
legend('Torque','Location','southeast');
xlabel('Angular Displacement [deg]')
ylabel('Torque [N-m]')

%Dynamic Graphs
```

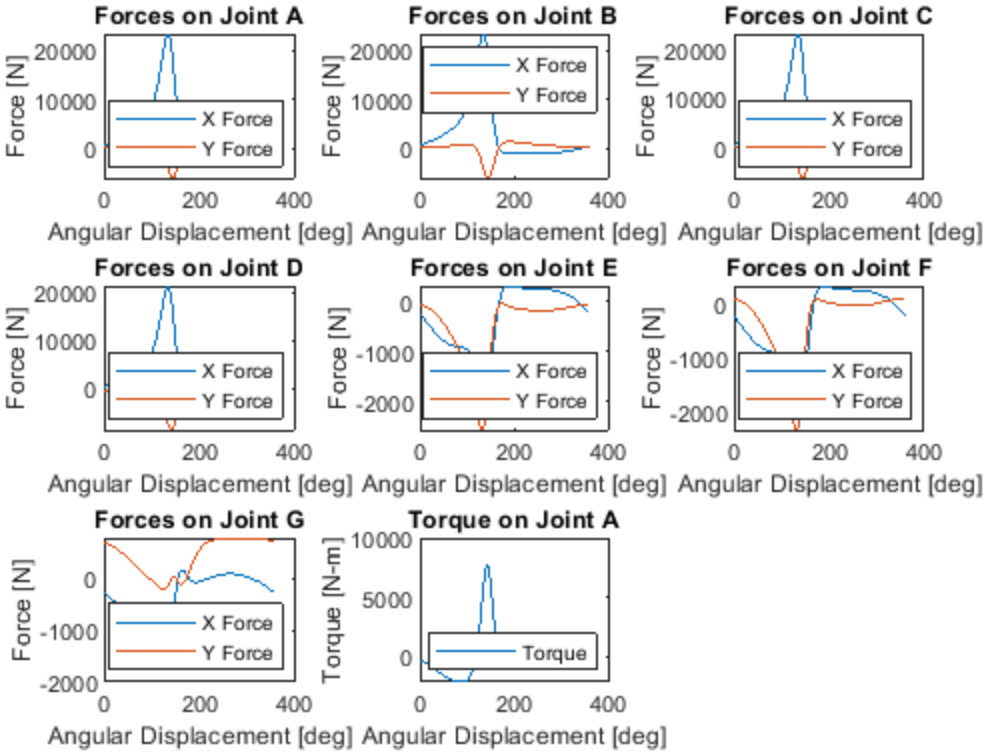
```
figure('name','Dynamic Joint Forces/Torque');
ax1= subplot(3,3,1);
plot(theta,dynamics(1,:),theta,dynamics(2,:));
title(ax1,'Forces on Joint A')
legend('X Force','Y Force','Location','southeast');
xlabel('Angular Displacement [deg]')
ylabel('Force [N]')
ax1= subplot(3,3,2);
plot(theta,dynamics(3,:),theta,dynamics(4,:));
title(ax1,'Forces on Joint B')
legend('X Force','Y Force','Location','northeast');
xlabel('Angular Displacement [deg]')
ylabel('Force [N]')
ax1= subplot(3,3,3);
plot(theta,dynamics(5,:),theta,dynamics(6,:));
title(ax1,'Forces on Joint C')
legend('X Force','Y Force','Location','southeast');
xlabel('Angular Displacement [deg]')
ylabel('Force [N]')
ax1= subplot(3,3,4);
plot(theta,dynamics(7,:),theta,dynamics(8,:));
title(ax1,'Forces on Joint D')
legend('X Force','Y Force','Location','southeast');
xlabel('Angular Displacement [deg]')
ylabel('Force [N]')
ax1= subplot(3,3,5);
plot(theta,dynamics(9,:),theta,dynamics(10,:));
title(ax1,'Forces on Joint E')
legend('X Force','Y Force','Location','southeast');
xlabel('Angular Displacement [deg]')
ylabel('Force [N]')
ax1= subplot(3,3,6);
plot(theta,dynamics(11,:),theta,dynamics(12,:));
title(ax1,'Forces on Joint F')
legend('X Force','Y Force','Location','southeast');
xlabel('Angular Displacement [deg]')
ylabel('Force [N]')
ax1= subplot(3,3,7);
plot(theta,dynamics(13,:),theta,dynamics(14,:));
title(ax1,'Forces on Joint G')
legend('X Force','Y Force','Location','southeast');
xlabel('Angular Displacement [deg]')
ylabel('Force [N]')
ax1= subplot(3,3,8);
plot(theta,dynamics(15,:));
title(ax1,'Torque on Joint A')
legend('Torque','Location','southeast');
xlabel('Angular Displacement [deg]')
ylabel('Torque [N-m]')

%Angular Accel/Velocity Graphs
figure('name','Angular Accelerations and Velocities');

%Plot velocities
```

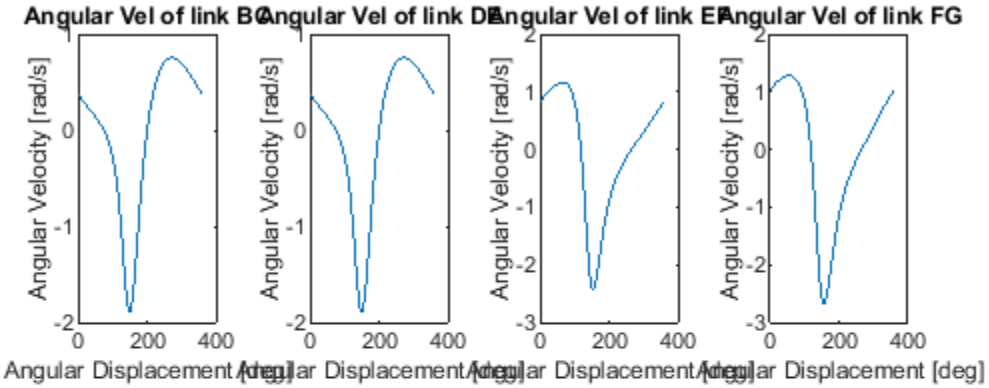
```
ax1= subplot(2,4,1);
plot(theta,angulars(1,:));
title(ax1,'Angular Vel of link BC')
xlabel('Angular Displacement [deg]')
ylabel('Angular Velocity [rad/s]')
ax1= subplot(2,4,2);
plot(theta,angulars(1,:));
title(ax1,'Angular Vel of link DE')
xlabel('Angular Displacement [deg]')
ylabel('Angular Velocity [rad/s]')
ax1= subplot(2,4,3);
plot(theta,angulars(3,:));
title(ax1,'Angular Vel of link EF')
xlabel('Angular Displacement [deg]')
ylabel('Angular Velocity [rad/s]')
ax1= subplot(2,4,4);
plot(theta,angulars(4,:));
title(ax1,'Angular Vel of link FG')
xlabel('Angular Displacement [deg]')
ylabel('Angular Velocity [rad/s]')

%Plot accelerations
ax1= subplot(2,4,5);
plot(theta,angulars(5,:));
title(ax1,'Angular Accel of link BC')
xlabel('Angular Displacement [deg]')
ylabel('Angular Acceleration [rad/s^2]')
ax1= subplot(2,4,6);
plot(theta,angulars(6,:));
title(ax1,'Angular Accel of link DE')
xlabel('Angular Displacement [deg]')
ylabel('Angular Acceleration [rad/s^2]')
ax1= subplot(2,4,7);
plot(theta,angulars(7,:));
title(ax1,'Angular Accel of link EF')
xlabel('Angular Displacement [deg]')
ylabel('Angular Acceleration [rad/s^2]')
ax1= subplot(2,4,8);
plot(theta,angulars(8,:));
title(ax1,'Angular Accel of link FG')
xlabel('Angular Displacement [deg]')
ylabel('Angular Acceleration [rad/s^2]')

%Linear Accel/Velocity Graphs
figure('name','Linear Accelerations and Velocities');
ax1= subplot(2,4,1);
plot(theta,angulars(9,:));
title(ax1,'Linear Vel of Joint B')
xlabel('Angular Displacement [deg]')
ylabel('Linear Velocity [m/s]')
ax1= subplot(2,4,2);
plot(theta,angulars(10,:));
title(ax1,'Linear Vel of Joint C')
xlabel('Angular Displacement [deg]')
```

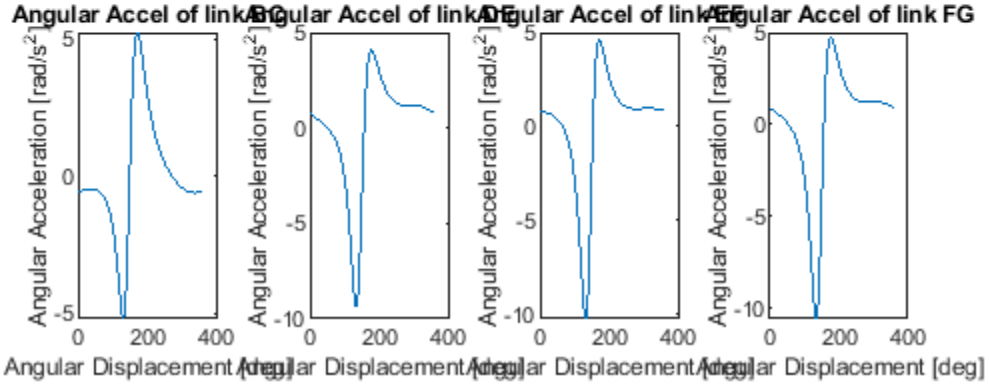
```
ylabel('Linear Velocity [m/s]')
ax1= subplot(2,4,3);
plot(theta,angulars(11,:));
title(ax1,'Linear Vel of Joint E')
xlabel('Angular Displacement [deg]')
ylabel('Linear Velocity [m/s]')
ax1= subplot(2,4,4);
plot(theta,angulars(12,:));
title(ax1,'Linear Vel of Joint F')
xlabel('Angular Displacement [deg]')
ylabel('Linear Velocity [m/s]')

%Plot acceleration
ax1= subplot(2,4,5);
plot(theta,angulars(13,:));
title(ax1,'Linear Accel of Joint B')
xlabel('Angular Displacement [deg]')
ylabel('Linear Acceleration [m/s^2]')
ax1= subplot(2,4,6);
plot(theta,angulars(14,:));
title(ax1,'Linear Accel of Joint C')
xlabel('Angular Displacement [deg]')
ylabel('Linear Acceleration [m/s^2]')
ax1= subplot(2,4,7);
plot(theta,angulars(15,:));
title(ax1,'Linear Accel of Joint E')
xlabel('Angular Displacement [deg]')
ylabel('Linear Acceleration [m/s^2]')
ax1= subplot(2,4,8);
plot(theta,angulars(16,:));
title(ax1,'Linear Accel of Joint F')
xlabel('Angular Displacement [deg]')
ylabel('Linear Acceleration [m/s^2]')

end
```

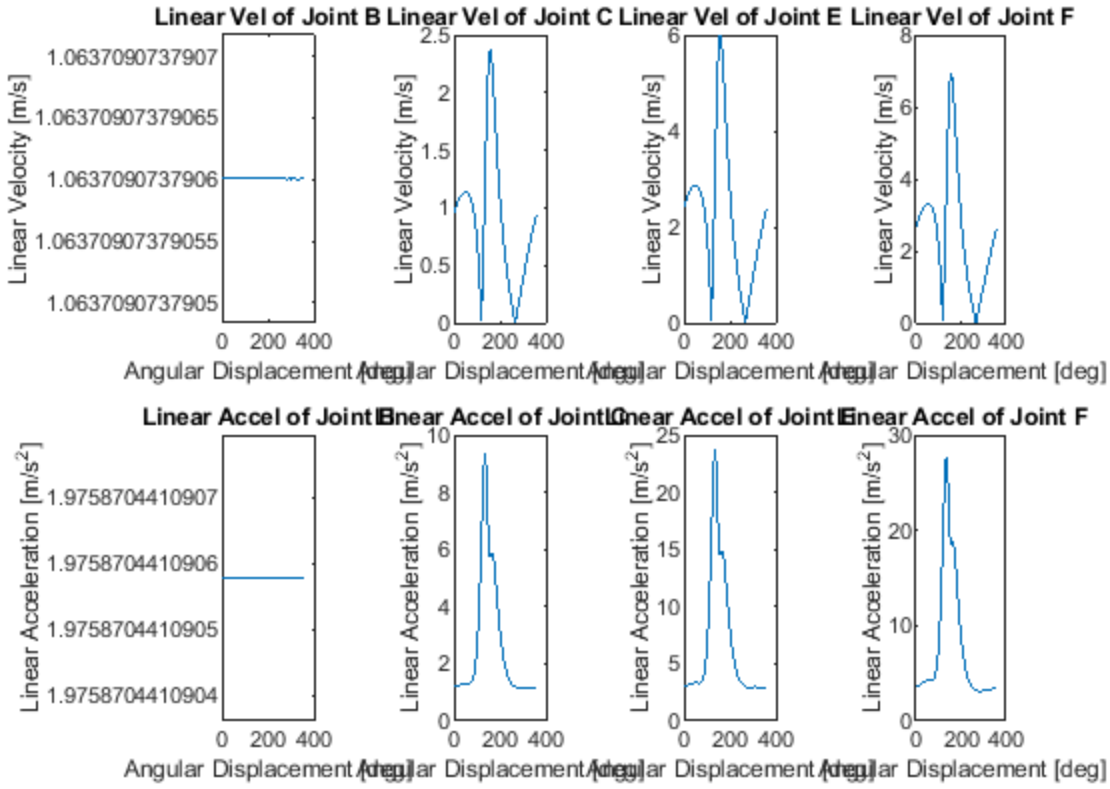

Joint B
Y Position [m]
X Position [m]
Joint C
Y Position [m]
X Position [m]
Joint E
Y Position [m]
X Position [m]
Joint F
Y Position [m]
X Position [m]


Forces on Joint A
Forces on Joint B
Forces on Joint C
Forces on Joint D
Forces on Joint E
Forces on Joint F
Forces on Joint G
Torque on Joint A
Force [N]
Torque [N-m]
Angular Displacement [deg]
X Force
Y Force
Torque


Forces on Joint A
Forces on Joint B
Forces on Joint C
Forces on Joint D
Forces on Joint E
Forces on Joint F
Forces on Joint G
Torque on Joint A
Force [N]
Torque [N-m]
X Force
Y Force
Torque
Angular Displacement [deg]


Angular Vel of link BC
Angular Vel of link DE
Angular Vel of link EF
Angular Vel of link FG
Angular Velocity [rad/s]
Angular Displacement [deg]


Angular Accel of link BC
Angular Accel of link DE
Angular Accel of link EF
Angular Accel of link FG
Angular Acceleration [rad/s^2]
Angular Displacement [deg]



Published with MATLAB® R2020a