Table of Contents


```
clear all
close all
clc
```

Initialize the system

```
par=[];
par.EOM=0;%flag for EOM deriviation
par.c_plot_figure=0;%flag for plot
exp_case=1;% 1:Step response 2-2-2 to 25-2-2 to 2-25-25 five times
%2: Step response 25-2-2 to 25-25-2 to 25-2-2 to 25-2-25 five times
par=func_exp_04_21(par,exp_case);
par.trianlge_length=70*1e-03;% fabric triangle edge length
par.L=0.19;%actuator length
par.n=8;% # of joints for augmented rigid arm
par.m0=0.35;%kg segment weight
par.g=9.8;%% gravity constant
par.a0=15*1e-03;
par.r_f=sqrt(3)/6*par.trianlge_length+par.a0; % we assume the force
 are evenly spread on a cirlce with radius of r_f
par.p1_angle=-30;%deg p1 position w/ the base frame
% update force position of p1 p2 and p3
for i =1:3

 par.r_p{i}=[par.r_f*cosd(par.p1_angle-120*(i-1)),par.r_f*sind(par.p1_angle-120*(i
    par.f_p{i}=588.31*par.pm_MPa(:,i+1);
end

exp. data...
```

Symbolic EOM

```
if par.EOM==1
par=func_EOM_v_1_3_2(par);
end
```

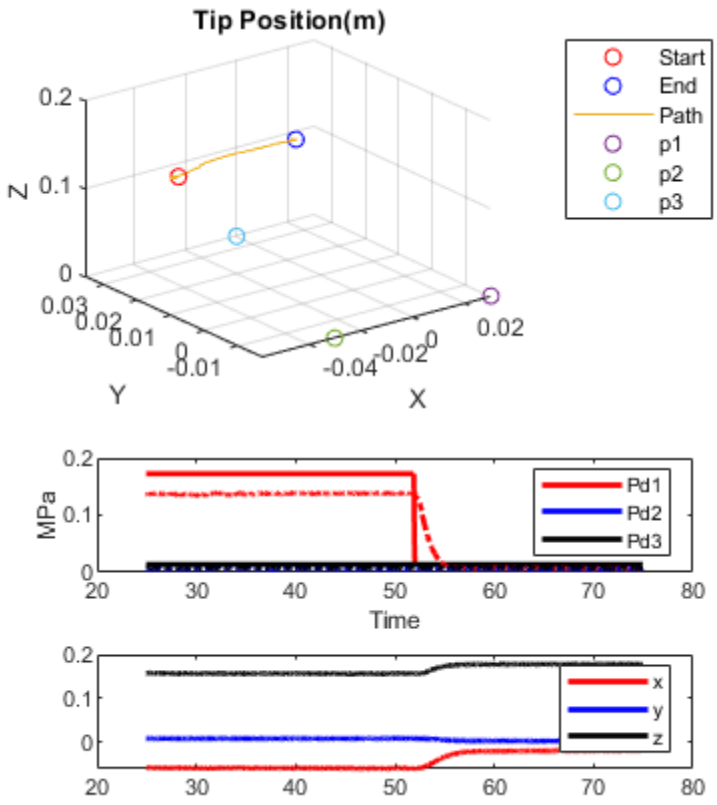
Get sample from the exp. data

```
sample=[];
temp.start=500;
temp.end=1500;
sample.pd_psi=par.pd_psi(temp.start:temp.end,1:end);
sample.pm_psi=par.pm_psi(temp.start:temp.end,1:end);
sample.pd_MPa=par.pd_MPa(temp.start:temp.end,1:end);
sample.pm_MPa=par.pm_MPa(temp.start:temp.end,1:end);
sample.tip_exp=par.tip_exp(temp.start:temp.end,1:end);
sample.r_p=par.r_p;
```

Plot Sample data

```
func_plot_sample_results(sample)
```

```
Plot Samples ...
```



Calculate Phi from the data

% Rotate w/ z-axis 0 deg

```
par.Rz_angle=0;%deg
par.Rz=[cosd(par.Rz_angle) sind(par.Rz_angle) 0;
        -sind(par.Rz_angle) cosd(par.Rz_angle) 0;
        0 0 1];
temp.phi_vector=sample.tip_exp(:,2:4); %xyz in Global frame
temp.angle_phi=[];
% Calculate phi anlge ranging [0,2pi] atan(y_top/x_top)
for i =1:length(sample.pd_psi)

 temp.angle_phi(i,1)=rad2deg(func_myatan(temp.phi_vector(i,2),temp.phi_vector(i,1)
end
sample.phi=temp.angle_phi;
sample.phi_rad=deg2rad(temp.angle_phi);
sample.phi_vector=temp.phi_vector;
```

XYZ in new frame (Z-Axis Rotation with phi)

```
tip_segi(:,1)=sample.tip_exp(:,1);
for i =1 : length(temp.phi_vector)
    temp.Rz=[];
    temp.Rz=[cosd(temp.angle_phi(i)) sind(temp.angle_phi(i)) 0;
        -sind(temp.angle_phi(i)) cosd(temp.angle_phi(i)) 0;
        0 0 1];
    temp.new_xyz=temp.phi_vector(i,1:3)*temp.Rz';
    tip_segi(i,2:4)=temp.new_xyz;
end
sample.tip_segi=tip_segi;
```

Theta calculation in X-Z plane no beta offset

```
theta_vector=[sample.tip_segi(:,2),sample.tip_segi(:,4)]; %xyz in
 Segittal frame z/x'
angle_haf_theta=90-atan2d(theta_vector(:,2),theta_vector(:,1));
sample.theta=2*angle_haf_theta;
sample.theta_rad=deg2rad(2*angle_haf_theta);
```

Inextensible layer on the Edge

find points (x_e,y_e) on three edges where (x_e,y_e)=beta*(x_top,y_top) beta <0 and min(beta)

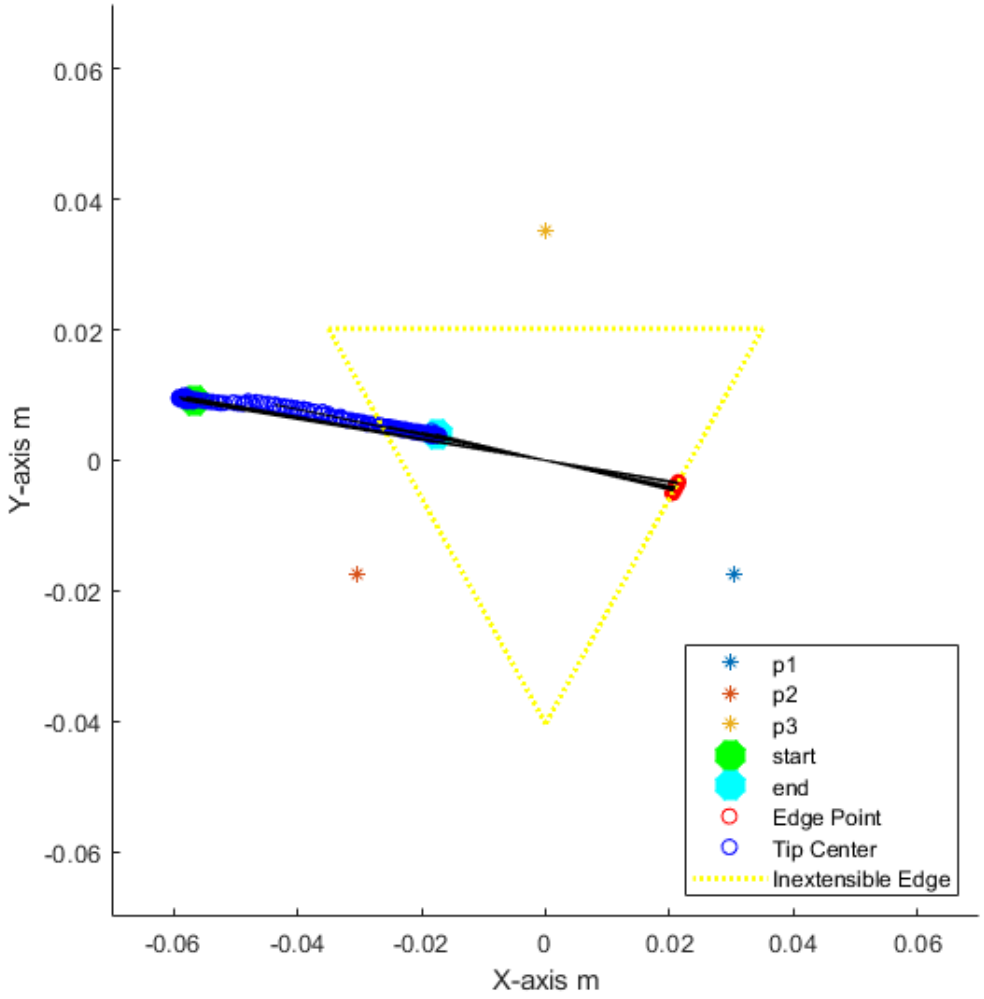
```
beta_array=zeros(length(sample.phi),3);
beta_array(:,1)=-par.trianlge_length./(sqrt(3)*(temp.phi_vector(:,2)-
sqrt(3)*temp.phi_vector(:,1)));
beta_array(:,2)=-par.trianlge_length./
(sqrt(3)*(temp.phi_vector(:,2)+sqrt(3)*temp.phi_vector(:,1)));
beta_array(:,3)=sqrt(3)*par.trianlge_length./(6*temp.phi_vector(:,2));
sample.beta=[];
for i=1:length(beta_array)% find beta <0 and min(beta)
```

```
    temp_array_1=[];temp_array_2=[];k=1;
    temp_array_1=beta_array(i,:);
    for j=1:length(temp_array_1)
        if temp_array_1(j)<=0
            temp_array_2(k)=temp_array_1(j);
            k=k+1;
        end
    end
        [beta_value,beta_pos]=min(abs(temp_array_2),[],2);
        sample.beta(i,1)=norm(beta_value*temp.phi_vector(i,1:2));
        sample.x_y_edge(i,1:2)=-beta_value*temp.phi_vector(i,1:2);
end
```

Plot Edege position

```
x1=linspace(0,0.035,100);
x2=linspace(-0.035,0,100);
x3=linspace(-0.035,0.035,100);
y1=sqrt(3)*x1-1/sqrt(3)*par.trianlge_length;
y2=-sqrt(3)*x2-1/sqrt(3)*par.trianlge_length;
y3=ones(length(x3),1)*sqrt(3)/6*par.trianlge_length;
% rotate with Rz
for i=1:length(x1)
    edge1(:,i)=par.Rz(1:2,1:2)*[x1(i);y1(i)];
    edge2(:,i)=par.Rz(1:2,1:2)*[x2(i);y2(i)];
    edge3(:,i)=par.Rz(1:2,1:2)*[x3(i);y3(i)];
end
par.c_plot_figure=1;
if par.c_plot_figure==1
    figure('Position',[100;100;600;600])
    for i =1:3
        scatter(sample.r_p{i}(1),sample.r_p{i}(2),'*')
    hold on
    end

 scatter(sample.phi_vector(1,1),sample.phi_vector(1,2),'g','*','LineWidth',12)
    hold on

 scatter(sample.phi_vector(end,1),sample.phi_vector(end,2),'c','*','LineWidth',12)
    hold on
    scatter(sample.x_y_edge(:,1),sample.x_y_edge(:,2),'r')
    hold on
    scatter(sample.phi_vector(:,1),sample.phi_vector(:,2),'b')
    hold on

 plot(edge1(1,:),edge1(2,:),'LineStyle',':','Color','y','LineWidth',2)
    hold on

 plot(edge2(1,:),edge2(2,:),'LineStyle',':','Color','y','LineWidth',2)
    hold on

 plot(edge3(1,:),edge3(2,:),'LineStyle',':','Color','y','LineWidth',2)
    hold on
```

```
    for i =1:20:length(sample.phi_vector)
        temp_v=[];
        temp_v=[sample.phi_vector(i,1:2);sample.x_y_edge(i,1:2)];
        plot(temp_v(:,1),temp_v(:,2),'k')
        hold on
    end

    legend('p1','p2','p3','start','end','Edge Point','Tip
 Center','Inextensible Edge','Location','southeast')
    xlabel('X-axis m')
    ylabel('Y-axis m')
    xlim([-0.07,0.07])
    ylim([-0.07,.07])
end
```



Augmented Rigid tip Estimation maps to Eq. (4) in the paper

```
xi_vector=[sample.phi_rad, sample.theta_rad/2, (par.L./
sample.theta_rad-sample.beta).*sin(sample.theta_rad/2),-
sample.phi_rad,...
    sample.phi_rad,(par.L./sample.theta_rad-
sample.beta).*sin(sample.theta_rad/2),sample.theta_rad/2, -
sample.phi_rad];
% xi_vector=[deg2rad(sample.phi), deg2rad(sample.theta/2), (par.L./
deg2rad(sample.theta)-sample.beta).*sind(sample.theta/2),...
%     (par.L./deg2rad(sample.theta)-
sample.beta).*sind(sample.theta/2), deg2rad(sample.theta/2), -
deg2rad(sample.phi)];
for j=1:length(xi_vector)

    xi=xi_vector(j,:);

 rigid_b0=sample.beta(j);thetad=sample.theta_rad(j);L=par.L;a0=par.a0;

 p1=sample.pm_MPa(j,2);p2=sample.pm_MPa(j,3);p3=sample.pm_MPa(j,4);
% Maps to Table 1. DH parameters
    rigid_a=zeros(1,par.n);
    rigid_alpha=[-pi/2 pi/2    0     0      0  -pi/2 pi/2      0];
    rigid_d=    [0      0   xi(3)    0      0  xi(6)    0      0];
    rigid_theta=[xi(1) xi(2)   0  xi(4)  xi(5)    0  xi(7) xi(8)];
    m=[0 0  par.m0];
% Get HTM T{i}, p{i}, z{i}, for Jacobian calculation
    Ti = cell(par.n+1,1);
    Ti(1) = {[1 0 0 0;0 1 0 0; 0 0 1 0; 0 0 0 1]};
    p_i{1}=Ti{1}(1:3,4);
    z_i{1}=Ti{1}(1:3,3);
for i = 2:par.n+1
    Ti{i} =  Ti{i-1} * ([cos(rigid_theta(i-1)) -sin(rigid_theta(i-1))
 0 0; sin(rigid_theta(i-1)) cos(rigid_theta(i-1)) 0 0; 0 0
 1 0; 0 0 0 1] *[1 0 0 0; 0 1 0 0; 0 0 1 rigid_d(i-1); 0 0
 0 1]*[1 0 0 rigid_a(i-1); 0 1 0 0; 0 0 1 0; 0 0 0 1]*[1 0
 0 0; 0 cos(rigid_alpha(i-1)) -sin(rigid_alpha(i-1)) 0 ; 0
 sin(rigid_alpha(i-1)) cos(rigid_alpha(i-1)) 0; 0 0 0 1]);
    p_i{i}=Ti{i}(1:3,4);
    z_i{i}=Ti{i}(1:3,3);
end
```

Linear Velocity J_v

```
J_v=cell(par.n,1);
i=par.n;
    j_v=(zeros(3,par.n));
    for j_counter =1:i
        if rigid_theta(j_counter) == 0 %% Prismatic Joint
            j_v(:,j_counter)=z_i{j_counter};
```

```
        else %% Rotational Joint
            j_v(:,j_counter)=cross(z_i{j_counter},(p_i{i+1}-
p_i{j_counter}));
        end
    end
    J_v{i}=j_v;
```

Angular Velocity J_w

```
J_w=cell(par.n,1);
i = par.n;
    j_w=(zeros(3,par.n));
    for j_counter =1:i
        if rigid_theta(j_counter) == 0 %% Prismatic Joint
            j_w(:,j_counter)=zeros(3,1);
        else %% Rotational Joint
            j_w(:,j_counter)=z_i{j_counter};
        end
    end
    J_w{i}=j_w;
```

Jacobian maps to Eq.(8)

```
J_xi2q{j}=[1, 0, 0,-1,1, 0, 0, -1;
           0, 0.5, - (cos(thetad/2)*(rigid_b0 - L/thetad))/2 -
 (L*sin(thetad/2))/thetad^2,0,0,  - (cos(thetad/2)*(rigid_b0 - L/
thetad))/2 - (L*sin(thetad/2))/thetad^2, 0.5, 0]';

J_x2xi{j}=[J_v{end};J_w{end}];
xyz_estimation(j,1:3)=(Ti{end}(1:3,4))';
```

Wrech force w/ base frame

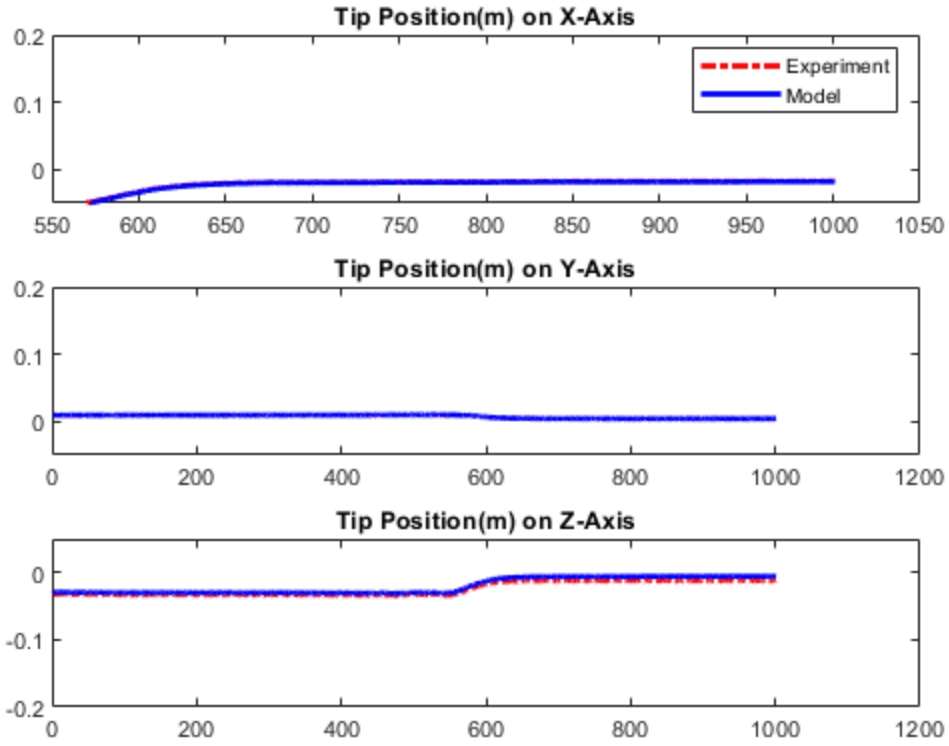
```
f_p1=588.31*p1;%mpa
f_p2=588.31*p2;%mpa
f_p3=588.31*p3;%mpa
for i =1:3
    T_p{i}=Ti{end}*[eye(3),sample.r_p{i};0 0 0 1];
    r_p_base{i}=T_p{i}(1:3,4);
end

sample.top_p1_xyz(j,1:3)=r_p_base{1};
sample.top_p2_xyz(j,1:3)=r_p_base{2};
sample.top_p3_xyz(j,1:3)=r_p_base{3};

wrench_base_pm{j}=[f_p1*T_p{1}(1:3,3);cross(r_p_base{1},f_p1*T_p{1}
(1:3,3))]+...
    [f_p2*T_p{2}(1:3,3);cross(r_p_base{2},f_p2*T_p{2}(1:3,3))]+...
    [f_p3*T_p{3}(1:3,3);cross(r_p_base{3},f_p3*T_p{3}(1:3,3))];

end
```

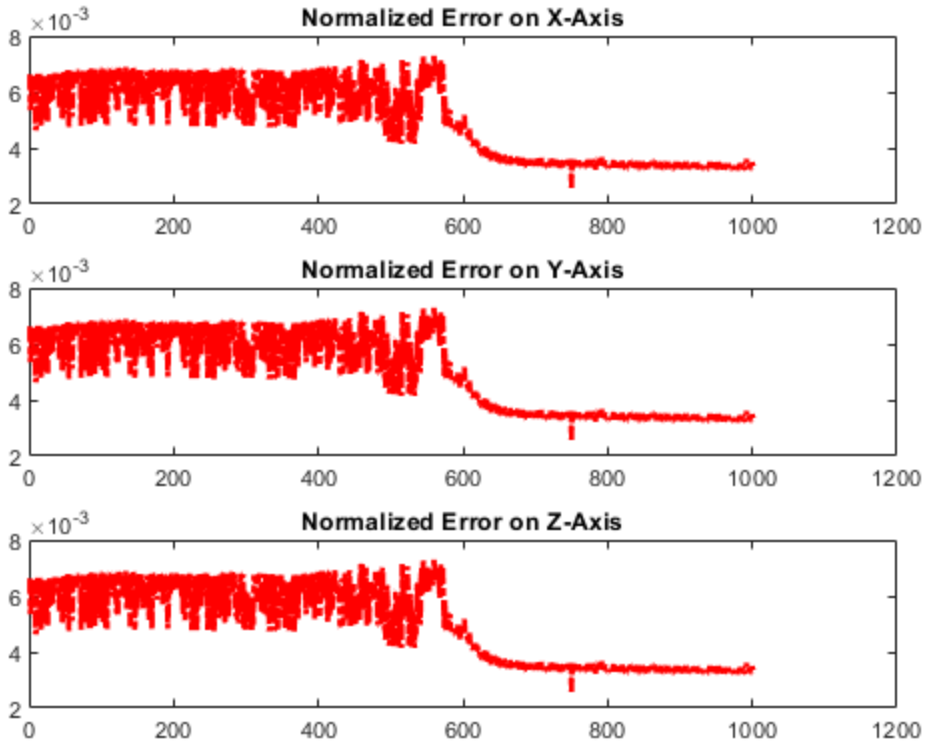
```
func_compare_kinematic(temp.phi_vector,xyz_estimation,par)
```

```
RMSE_x =

   9.8353e-04
```

```
RMSE_y =

   1.6932e-04
```

```
RMSE_z =

    0.0046
```





Project froce in base frame to joint frame and arc frame

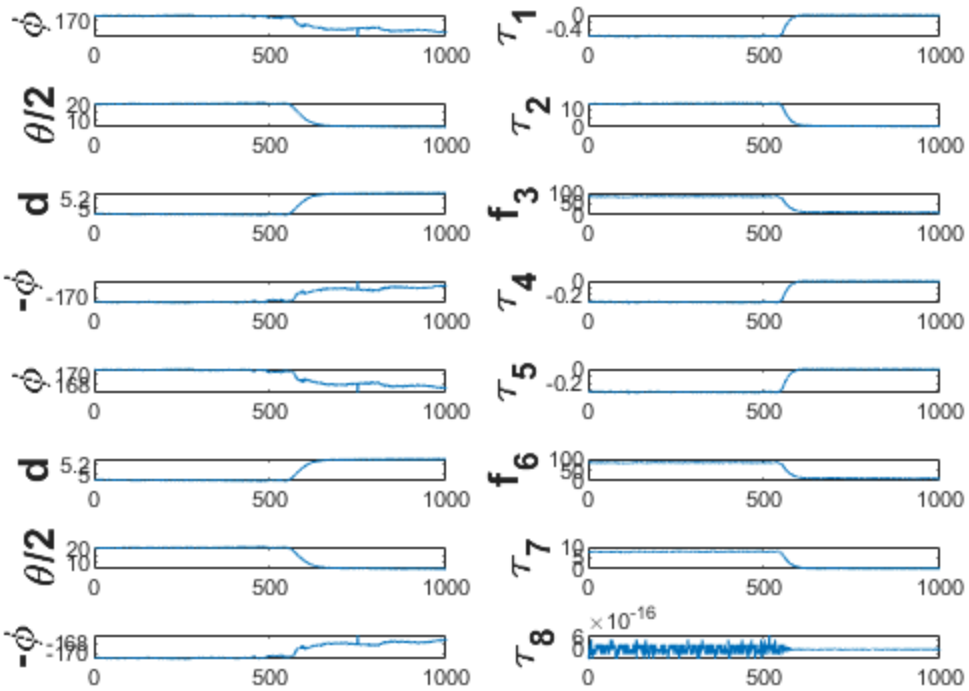
Wrech Force for sample maps to Eq. (10) and (11)

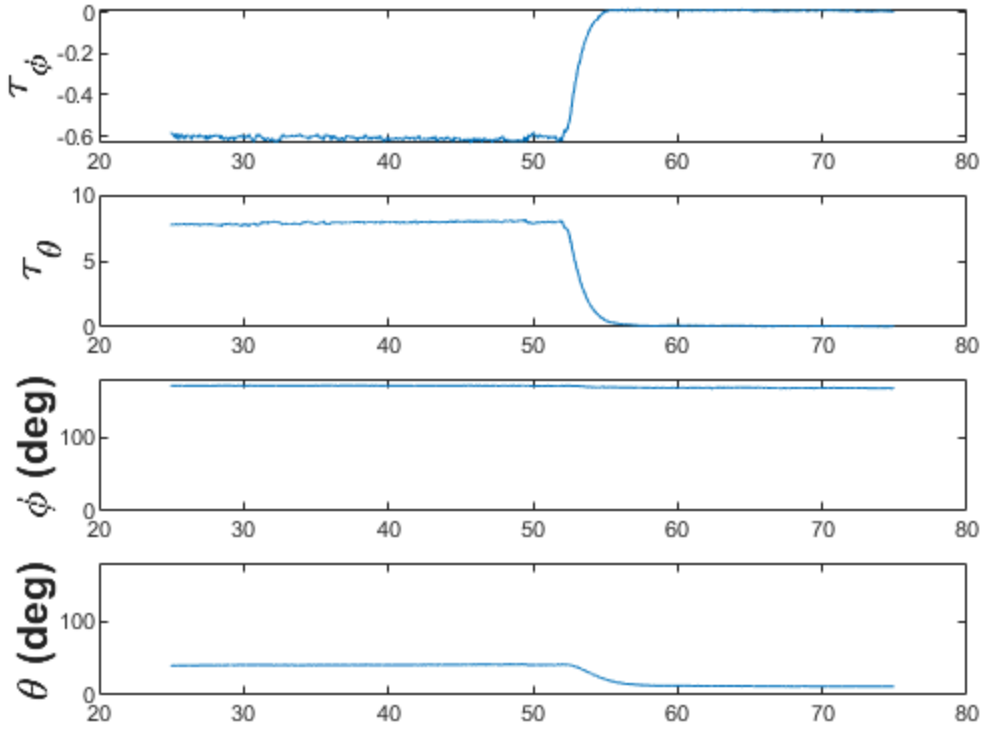
```
for i =1:length(sample.pm_MPa)
   temp_torque(1:2,i)=J_xi2q{i}'*J_x2xi{i}'*wrench_base_pm{i};
   temp_tau_xi(:,i)=J_x2xi{i}'*wrench_base_pm{i};
   temp_tau_base_frame(:,i)=wrench_base_pm{i};
end
sample.tau_phi=temp_torque(1,:);
sample.tau_theta=temp_torque(2,:);
sample.tau_xi=temp_tau_xi;
sample.tau_base_frame=temp_tau_base_frame;
```

Plot result

```
temp_figure=figure(6);
label_y_array={'\phi','\theta/2','d','-\phi','\phi','d','\theta/2','-
\phi'};
label_y_array2={'\tau_1','\tau_2','f_3','\tau_4','\tau_5','f_6','\tau_7','\tau_8'}
for i =1:2:par.n*2
subplot(par.n,2,i)
plot(rad2deg(xi_vector(:,(i+1)/2)))
```

```
ylabel(label_y_array{(i+1)/2},'FontWeight','bold','FontSize',16)
subplot(par.n,2,i+1)
plot(sample.tau_xi((i+1)/2,:))
ylabel(label_y_array2{(i+1)/2},'FontWeight','bold','FontSize',16)
end
temp_figure=figure(7);
label_y_array={'\tau_{\phi}','\tau_{\theta}'};
for i =1:2
subplot(4,1,i)
plot(sample.pd_psi(:,1),temp_torque(i,:))
ylabel(label_y_array{i},'FontWeight','bold','FontSize',16)
end
hold on
subplot(4,1,3)
plot(sample.pd_psi(:,1),sample.phi)
ylabel('\phi (deg)','FontWeight','bold','FontSize',16)
ylim([0 180])
hold on
subplot(4,1,4)
plot(sample.pd_psi(:,1),sample.theta)
ylabel('\theta (deg)','FontWeight','bold','FontSize',16)
ylim([0 180])
```





Published with MATLAB® R2018b