Table of Contents

Main function for stiffness ID use data 0704

```
clear all
close all
clc
```

Initialize the system

```
par_set=[];
%flag for EOM deriviation
par_set.EOM=1;
%flag for plot
par_set.flag_plot_rawData = 1;
%flag for read txt file or mat file 1: txt 0: mat
par_set.flag_read_exp = 0;
%flag for plotting moving constant layer
par_set.flag_plot_movingCC =0;
% p1 > p2,3
par_set.trial_2_25psi=[];
par_set.trial_5_25psi=[];
par_set.trial_1_25psi=[];
par_set.trial_0_25psi=[];
% Geometric para.
par_set.trianlge_length=70*1e-03;% fabric triangle edge length
par_set.L=0.19;%actuator length
par_set.n=4;% # of joints for augmented rigid arm
par_set.m0=0.35;%kg segment weight
par_set.g=9.8;%% gravity constant
par_set.a0=15*1e-03;%% 1/2 of pillow width
par_set.r_f=sqrt(3)/6*par_set.trianlge_length+par_set.a0; % we assume
 the force are evenly spread on a cirlce with radius of r_f
```

Update location of 3 chambers P1, P2, P3

```
par_set.p1_angle=-150;%deg p1 position w/ the base frame
% update force position of p1 p2 and p3
for i =1:3
    par_set.r_p{i}=[par_set.r_f*cosd(par_set.p1_angle
+120*(i-1)),par_set.r_f*sind(par_set.p1_angle+120*(i-1)),0].';
%     par_set.f_p{i}=588.31*par_set.pm_MPa(:,i+1);
end
fprintf('System initialization done \n')
```

```
System initialization done
```

Read txt file or mat file

```
if par_set.flag_read_exp==1

 par_set.trial_2_25psi=func_high_level_exp(par_set.trial_2_25psi,1);

 par_set.trial_5_25psi=func_high_level_exp(par_set.trial_5_25psi,2);

 par_set.trial_1_25psi=func_high_level_exp(par_set.trial_1_25psi,3);

 par_set.trial_0_25psi=func_high_level_exp(par_set.trial_0_25psi,4);
    save('raw_id_data.mat','par_set');
    fprintf( 'Saved \n' )
else
    fprintf( 'Loading... \n' );
    load('raw_id_data.mat');
    fprintf( 'Data loaded \n' );
end
```

```
Loading...
Data loaded
```

Get sample from the exp. data

```
trainSet=[];test_data=[];
test_data=par_set.trial_1_25psi;
[val,pos]=findpeaks(test_data.pd_psi(:,2));
trainSet.r_p=par_set.r_p;
%%%%%%%%%%%%%%%
trainSet.pd_psi=test_data.pd_psi(1:pos(6)-10,1:end);
trainSet.pm_psi=test_data.pm_psi(1:pos(6)-10,1:end);
trainSet.pd_MPa=test_data.pd_MPa(1:pos(6)-10,1:end);
trainSet.pm_MPa=test_data.pm_MPa(1:pos(6)-10,1:end);
trainSet.tip_exp=test_data.tip_exp(1:pos(6)-10,1:end);
%%%%%%%%%%%%
validSet.pd_psi=test_data.pd_psi(pos(6)-9,1:end);
validSet.pm_psi=test_data.pm_psi(pos(6)-9,1:end);
validSet.pd_MPa=test_data.pd_MPa(pos(6)-9,1:end);
validSet.pm_MPa=test_data.pm_MPa(pos(6)-9,1:end);
validSet.tip_exp=test_data.tip_exp(pos(6)-9,1:end);
fprintf('Dividing trainning set and validation set\n')
```

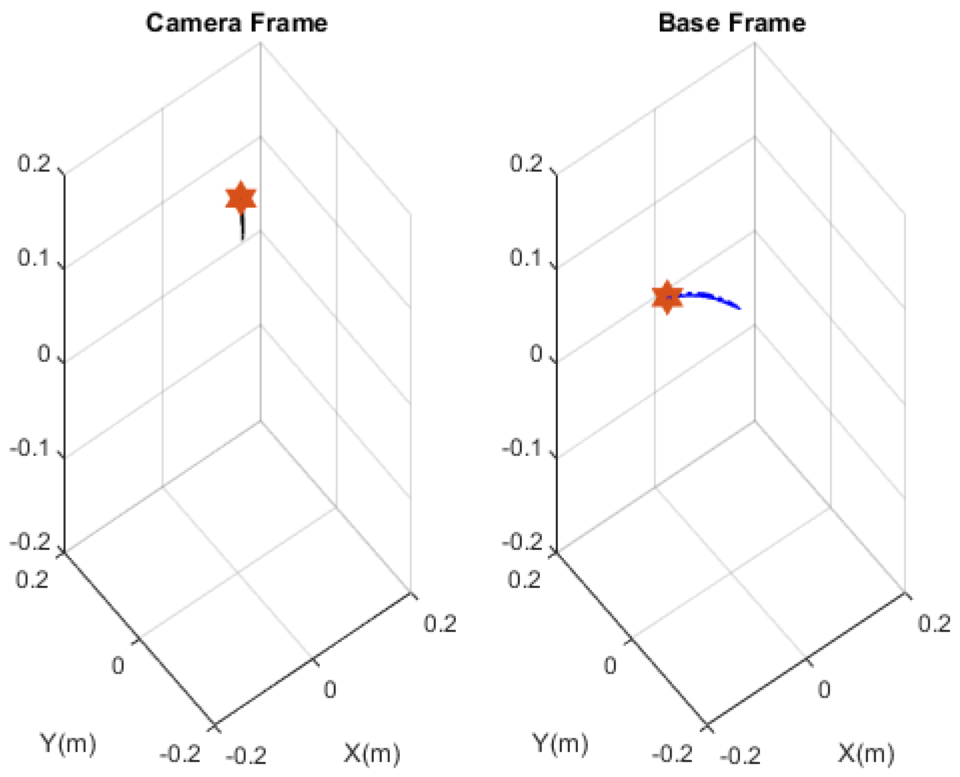
```
Dividing trainning set and validation set
```

Calculate Phi in camera frame then maps to robot base frame 0,2pi

```
temp.phi_vector=trainSet.tip_exp(:,2:4); %xyz in Camera frame
temp.angle_phi=[];
```

Calculate phi anlge ranging [0,2pi] atan(y_top/x_top)

```
temp.tip_exp_baseFrame(:,1)=trainSet.tip_exp(:,1);
for i =1:length(trainSet.pd_psi)

 temp.angle_phi(i,1)=rad2deg(func_myatan(temp.phi_vector(i,2),temp.phi_vector(i,1)
    temp.Rz=[cosd(temp.angle_phi(i,1)) -sind(temp.angle_phi(i,1))  0;
            sind(temp.angle_phi(i,1)) cosd(temp.angle_phi(i,1))   0;
            0                   0                   1];
 temp.Rz2=[1 0 0
            0 0 -1
            0 1 0];
temp.R_cam2Base=[cosd(-temp.angle_phi(i,1)) 0 sind(-
temp.angle_phi(i,1));
                 sind(-temp.angle_phi(i,1)) 0 -cosd(-
temp.angle_phi(i,1));
                    0                   1                   0];
%         temp.Rz2=eye(3);

 temp.tip_exp_baseFrame(i,2:4)=((temp.Rz*temp.Rz2)'*(trainSet.tip_exp(i,2:4)'))';
end
trainSet.phi=temp.angle_phi;
trainSet.phi_rad=deg2rad(temp.angle_phi);
trainSet.tip_exp_baseFrame=temp.tip_exp_baseFrame;
%%%%%%%
if par_set.flag_plot_rawData==1
func_compareCamWithBase(trainSet)
end
```



Calculate b_i in Camera frame

```
beta_array=zeros(length(trainSet.phi),3);
beta_array(:,1)=-par_set.trianlge_length./
(sqrt(3)*temp.phi_vector(:,2)+3*temp.phi_vector(:,1));
beta_array(:,2)=-par_set.trianlge_length./
(sqrt(3)*temp.phi_vector(:,2)-3*temp.phi_vector(:,1));
beta_array(:,3)=sqrt(3)*par_set.trianlge_length./
(6*temp.phi_vector(:,2));
r_beta=zeros(length(trainSet.phi),3);
trainSet.beta=[];
for i=1:length(beta_array)% find beta <0 and ||beta*(xt,yt)|| <= a0/
sqrt(3)
    beta_k=beta_array(i,:);
    trainSet.beta(i,1)=0;
    trainSet.x_y_edge(i,1:2)=[0,0];
    for k =1:3
        r_beta_k(i,k)=norm(beta_k(k).*trainSet.tip_exp(i,2:3));
        if beta_k(k)<0
            if  r_beta_k(i,k)<= par_set.trianlge_length/sqrt(3)
                temp.Rz=[cosd(temp.angle_phi(i,1)) -
sind(temp.angle_phi(i,1))  0;
                        sind(temp.angle_phi(i,1))
 cosd(temp.angle_phi(i,1))   0;
                        0                   0                   1];
```

```
%                  trainSet.beta(i,1)=r_beta_k(i,k);

 trainSet.x_y_edge(i,1:2)=beta_k(k)*trainSet.tip_exp(i,2:3);
                trainSet.x_y_edge(i,3)=0;
                temp_r=temp.Rz'*trainSet.x_y_edge(i,:)';
                trainSet.beta(i,1)=temp_r(1);
            end
        end
    end
end
%%%%%%
%%%%%%
if par_set.flag_plot_movingCC==1
   func_camFramePlotMovingCC(trainSet);
end
```

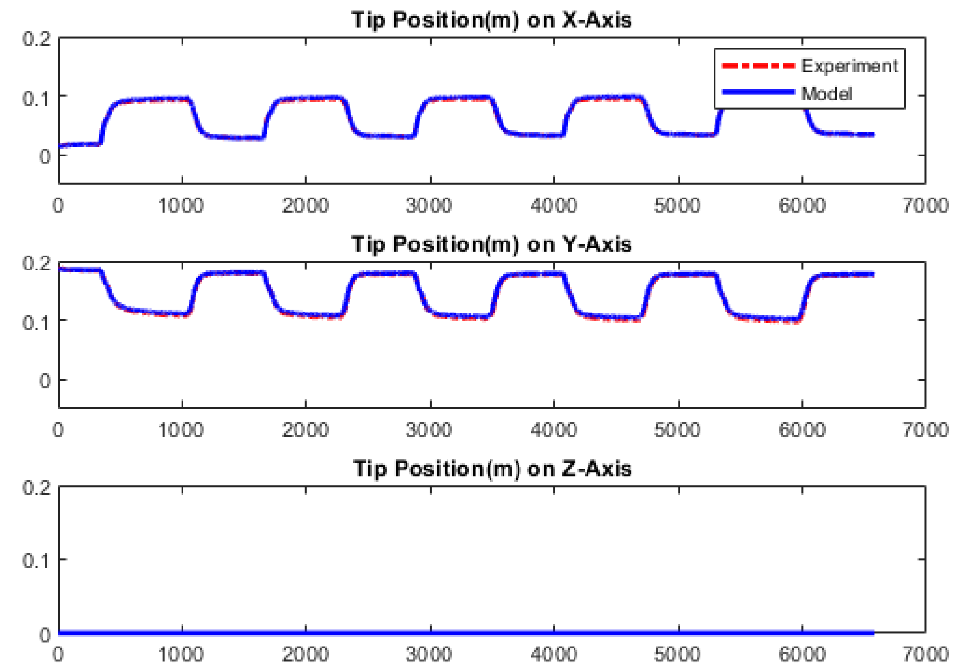
Calculate theta in base frame ranging -pi/2,pi/2

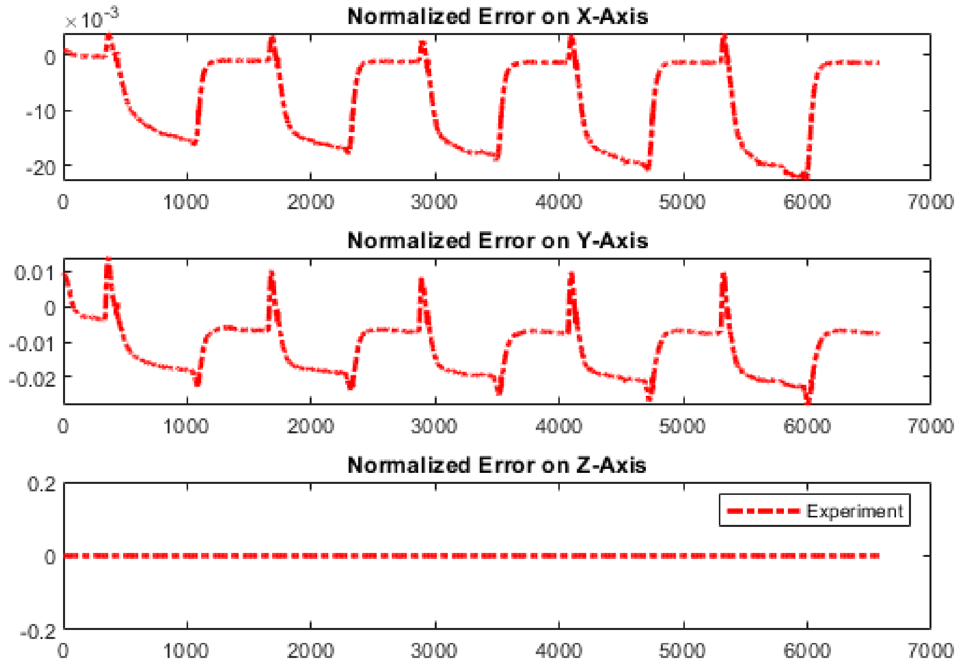
```
trainSet.theta_rad=2*-
sign(trainSet.tip_exp_baseFrame(:,2)).*asin(sqrt(trainSet.tip_exp_baseFrame(:,2).^
sqrt(trainSet.tip_exp_baseFrame(:,2).^2+trainSet.tip_exp_baseFrame(:,3).^2));
trainSet.theta_deg=rad2deg(trainSet.theta_rad);
```

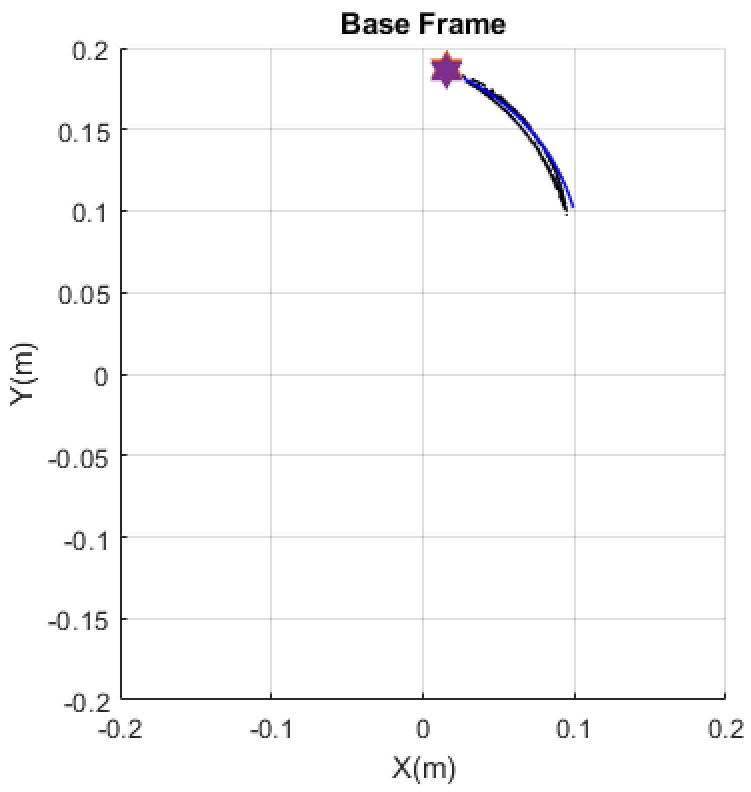
Augmented Rigid tip Estimation in Base frame

```
trainSet.xi_vector=[trainSet.theta_rad/2,(par_set.L./
trainSet.theta_rad-trainSet.beta).*sin(trainSet.theta_rad/2),
(par_set.L./trainSet.theta_rad-
trainSet.beta).*(sin(trainSet.theta_rad/2)),trainSet.theta_rad/2];

trainSet=func_fwdKinematic(trainSet,par_set);
func_compare_kinematic_YZ(trainSet,trainSet.xyz_estimation,par_set);
```

```

RMSE_x =

    0.0022


RMSE_y =

    0.0027


RMSE_z =

   1.0295e-17

Warning: Ignoring extra legend entries.
```


Tip Position(m) on X-Axis
Experiment
Model
Tip Position(m) on Y-Axis
Tip Position(m) on Z-Axis


Normalized Error on X-Axis
×10^-3
Normalized Error on Y-Axis
Normalized Error on Z-Axis
Experiment



Symbolic EOM

```
if par_set.EOM==1
par_set=func_EOM_baseFrame(par_set);
end
```

```
EOM...
EOM Done
```

Getting velocity and acceleration

5 sample moving average method is used

```
windowSize = 5;
filter_b = (1/windowSize)*ones(1,windowSize);
filter_a = 1;
trainSet.velocity_phi_rad=zeros(length(trainSet.phi_rad),1);
trainSet.velocity_phi_rad(2:end)=filter(filter_b,filter_a,
(trainSet.phi_rad(2:end)-trainSet.phi_rad(1:end-1))/(1/20));
trainSet.acc_phi_rad=zeros(length(trainSet.phi_rad),1);
trainSet.acc_phi_rad(2:end)=filter(filter_b,filter_a,
(trainSet.velocity_phi_rad(2:end)-trainSet.velocity_phi_rad(1:end-1))/
(1/20));

trainSet.velocity_theta_rad=zeros(length(trainSet.theta_rad),1);
trainSet.velocity_theta_rad(2:end)=smooth((trainSet.theta_rad(2:end)-
trainSet.theta_rad(1:end-1))/(1/20));
trainSet.acc_theta_rad=zeros(length(trainSet.theta_rad),1);
trainSet.acc_theta_rad(2:end)=smooth((trainSet.velocity_theta_rad(2:end)-
trainSet.velocity_theta_rad(1:end-1))/(1/20));
```

leat square estimation for alpha k d

```
m0=par_set.m0;L=par_set.L;g=9.8;
for i =1:length(trainSet.pd_MPa)
theta=trainSet.theta_rad(i);phi=trainSet.phi_rad(i);b0=trainSet.beta(i);
dtheta=trainSet.velocity_theta_rad(i);dphi=trainSet.velocity_phi_rad(i);
pm1=trainSet.pm_MPa(i,2);pm2=trainSet.pm_MPa(i,3);pm3=trainSet.pm_MPa(i,4);
Izz=m0*b0^2;
M(i,1)=Izz/4 + m0*((cos(theta/2)*(b0 - L/theta))/2 + (L*sin(theta/2))/
theta^2)^2 + (m0*sin(theta/2)^2*(b0 - L/theta)^2)/4;
C(i,1)= (dtheta*m0*sin(theta/2)*(b0 - L/theta)*((cos(theta/2)*(b0 -
 L/theta))/2 + (L*sin(theta/2))/theta^2))/4 - m0*((cos(theta/2)*(b0
 - L/theta))/2 + (L*sin(theta/2))/theta^2)*((dtheta*sin(theta/2)*(b0
 - L/theta))/4 - (L*dtheta*cos(theta/2))/theta^2 +
 (2*L*dtheta*sin(theta/2))/theta^3);
C_simp(i,1)=-(L*dtheta*m0*(2*sin(theta/2) -
 theta*cos(theta/2))*(2*L*sin(theta/2) - L*theta*cos(theta/2) +
 b0*theta^2*cos(theta/2)))/(2*theta^5);
G(i,1)=(g*m0*sin(theta/2)^2*(b0 - L/theta))/2 -
 g*m0*cos(theta/2)*((cos(theta/2)*(b0 - L/theta))/2 +
 (L*sin(theta/2))/theta^2);
G_simp(i,1)=-(g*m0*(L*sin(theta) + b0*theta^2*cos(theta) -
 L*theta*cos(theta)))/(2*theta^2);
pi_alpha(i,1)=sin(phi)*(0.5*pm1+0.5*pm2-pm3)-
sqrt(3)*cos(phi)*(0.5*pm1-0.5*pm2);
pi_k(i,1)=-theta;
pi_d(i,1)=-dtheta;
end
%%%%
tau_bar=M.*trainSet.acc_theta_rad+C_simp.*trainSet.velocity_theta_rad
+G_simp;
Y_bar=[pi_alpha,pi_k,pi_d];
%%%%
start_pt=2000;
end_pt=length(Y_bar);
%%%%
temp.result=inv(Y_bar(start_pt:end_pt,:).'*Y_bar(start_pt:end_pt,:))*Y_bar(start_p
fprintf('Estimated [alpha,k,d] is [%.4f,%.4f,%.4f]
 \n',temp.result(1),temp.result(2),temp.result(3))
```

```
Estimated [alpha,k,d] is [4.2570,0.1756,0.2664]
```

Published with MATLAB® R2018b