```matlab
%% HW6 Code for implementing Adaptive and Robust Controller on 1 Link Arm
%% By: Mostafa Atalla
%% RBE 502 Robotics Control course by Prof. Jie Fu

clc
clear all;
close all;


%% Initialization for Trajectory Tracking%%
% Setting the initial,final conditions and the time span.
x0=[-1 0.5];          %Setting initial conditions for the state vector
tf=5;                %Final time

%% Adaptive Passivity based Controller Input
lambda=1;                                  %Lambda Square positive definite matrix for
lyapanuv Based Controller.
kv=10;                                      %Kv matrix.
L=[0.6 0 0;0 0.1 0;0 0 0.3];               %Symmetric Positive definite matrix

cont_input={lambda,kv,L}                   %Controller input cell

%original system paramters%
I=7.5;
mgd=6;
fv=1.5;

original_param=[I mgd fv]; %original parameters vector

%Initial estimate%
I_e=8;
mgd_e=5;
fv_e=2.5;

estimate_param=[I_e mgd_e fv_e]; %estimated parameters vector

% Initial alfa
alfa0=[mgd_e; fv_e; I_e]
x0_p=[x0 mgd_e fv_e I_e];
%% Inverse Dynamics Robust Controller
gama=[10 10 10 10];
p=[1 0;0 1];
B=[0;1];
kp=40;
kd=30;

%% GENERATE TRAJECTORY USING TwoLinkArmTraj matlab file.
t=0:0.01:tf;
figure('Name','Position');
plot(t,-sin(t),'LineWidth',3);
title('Position (degree)')
grid on

figure('Name','Velocity');
plot(t,-cos(t),'LineWidth',3);
title('Velocity (degree/s)')
grid on


%nofigure=2;
%[a] = TwoLinkArmTraj(x0_traj(1),x0_traj(2), xf(1),xf(2),tf, nofigure) %Trajectory
Generation for the first Link
```

```matlab
%% Implement the adaptivePassive control
global ii
ii=1
global jj
jj=0
global ddtheta
ddtheta(ii)=0;
options = odeset('RelTol',1e-4,'AbsTol',[1e-4, 1e-4 1e-4, 1e-4 1e-4]);
[T,X] = ode45(@(t,x)
adaptivePassive(t,x,cont_input,original_param,estimate_param,alfa0),[0 tf],x0_p, options)

figure(1);
subplot
hold on
plot(T, X(:,1),'m-');
legend('Position Trajectory','Passivity Based Adaptive Controller')
title('Position Convergence Under Passivity Based Adaptive Controller')
xlabel('Time')
ylabel('Theta (Degrees)')

figure(2);
hold on
plot(T, X(:,2),'m--');
legend('Velocity Trajectory','Passivity Based Adaptive Controller')
title('Velociy Convergence Under Passivity Based Adaptive Controller')
xlabel('Time')
ylabel('Theta_dot')

%% error in the parameters
%error in the mgd
figure(3);
error_mgd=(abs(X(:,3)-mgd)/mgd)*100;
plot(T,error_mgd,'k--');
hold on
error_fv=(abs(X(:,4)-fv)/fv)*100;
plot(T,error_fv,'b-');
hold on
error_I=(abs(X(:,5)-I)/I)*100;
plot(T,error_I,'m-');
hold on
legend('error in gravity term','error in friction term','error in inertia term')
title('Errors in the parameters')
xlabel('Time')
ylabel('Error Percentege')
grid on
%% Inverse Dynamics Robust Controller
global o
options = odeset('RelTol',1e-4,'AbsTol',[1e-4, 1e-4]);
[T,Y] =
ode45(@(t,y)inverseDCRobust(t,y,gama,B,p,kp,kd,estimate_param,original_param,o),[0
tf],x0, options)%

figure(1);
hold on
plot(T, Y(:,1),'k--');
legend('Position Trajectory','Passivity Based Adaptive Controller','Inverse Dynamics
Robust Controller')
title('Position Convergence Under Inverse Dynamics Robust Controller')
xlabel('Time')
```

```
ylabel('Theta (Degrees)')

figure(2);
hold on
plot(T, Y(:,2),'k-');
legend('Velocity Trajectory','Passivity Based Adaptive Controller','Inverse Dynamics
Robust Controller')
title('Velocity Convergence Under Inverse Dynamics Robust Controller')
xlabel('Time')
ylabel('Theta_dot')
```