```
%% Inverse Dynamics Robust Controller
%% RBE 502 Fall 2018
%% Homework 6
%% December 6, 2018

function dydt = inverseDCRobust(t,y,gama,B,p,kp,kd,estimate_param,original_param,o)
%% Parameters of the original system


I=original_param(1,1);
mgd=original_param(1,2);
fv=original_param(1,3);

%% Parameters of the estimated model
I_e=estimate_param(1,1);
mgd_e=estimate_param(1,2);
fv_e=estimate_param(1,3);

%% Desired Trajectory setting up
theta_d=-sin(t);
dtheta_d=-cos(t);
ddtheta_d=sin(t);
 %% Controller setting up
ro=gama(1)*y(1)+gama(2)*y(2)+gama(3)*(y(2)^2)+gama(4)

 e=y(1)-theta_d          %position error vector
 e_dot=y(2)-dtheta_d     %Velocity error vector

 k=[e;e_dot]             %error state vector

 epsilon=1;
 if (B'*p*k)>epsilon
     v=(-B'*p*k*ro)/(B'*p*k);
 else
     v=(-B'*p*k*ro)/epsilon;
 end

 aq=ddtheta_d-kp*e-kd*e_dot+v;

 meo=((I_e/I)-1)*aq+((fv_e-fv)*y(2))/I+((mgd_e-mgd)*sin(y(1)))/I;

 if meo>ro
     o=1;
 else
     o=0;
 end
 figure(4)
 hold on
 grid on
 plot(t,o,'*')
 ylim([0 2])
 title('Points breaking the boundness')
 xlabel('Time')
 ylabel('Points breaking at value=1')
 hold on
 u= I_e*aq+fv_e*y(2)+mgd_e*sin(y(1));

 theta_dd=(u-mgd*sin(y(1))-fv*(y(2)))/I;


dydt=zeros(2,1)
dydt(1)=y(2);
```

```
dydt(2)= theta_dd;

end
```