PLOTS: SIGNUM FUNCTION

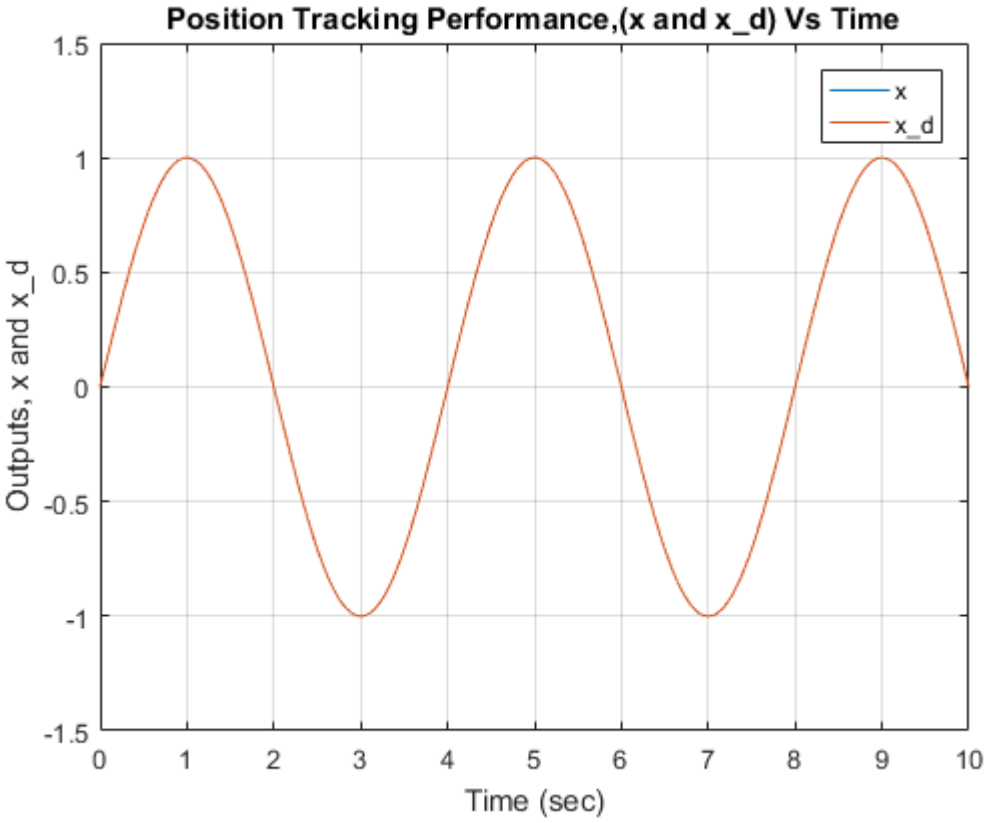

Figure 1: Position Tracking Performance

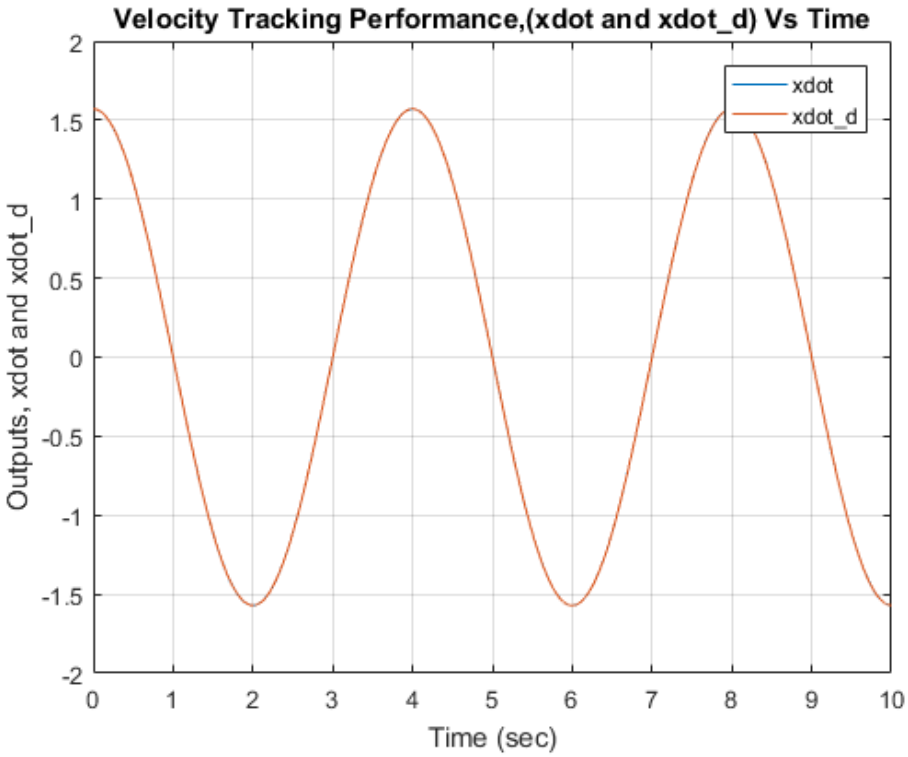

Figure 2: Velocity Tracking Performance

Figure 3: Position Tracking Error Performance

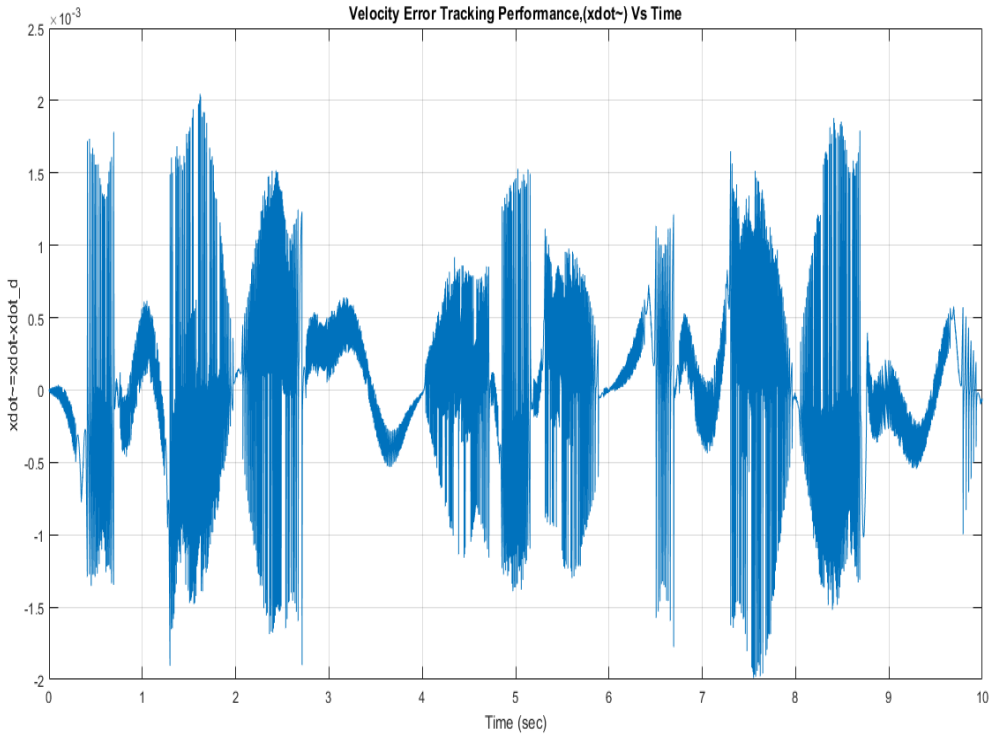

Figure 4: Velocity Tracking Error Performance

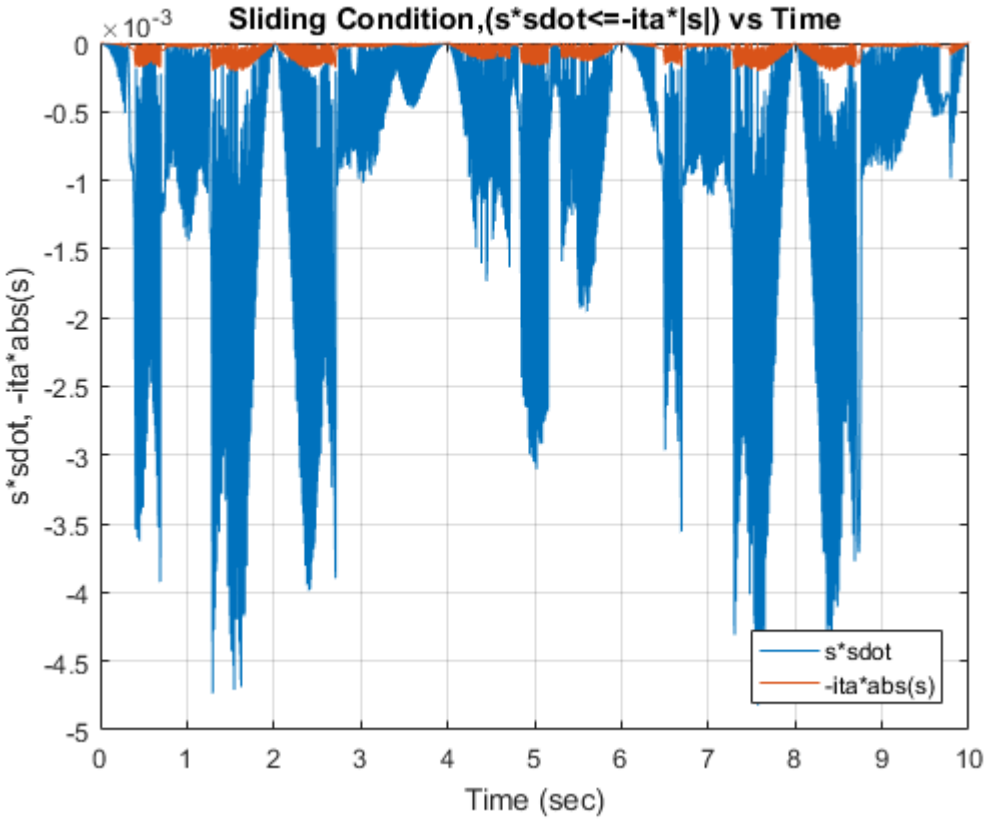

Figure 5: Sliding Condition

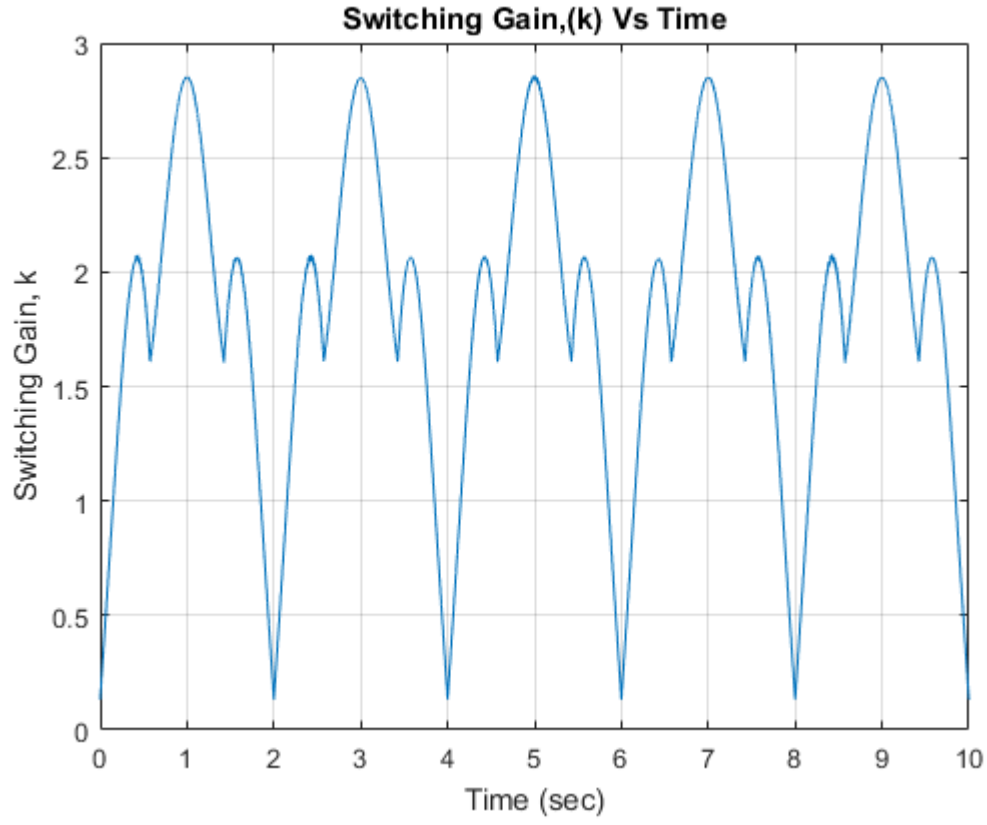

Figure 6: Switching Gain

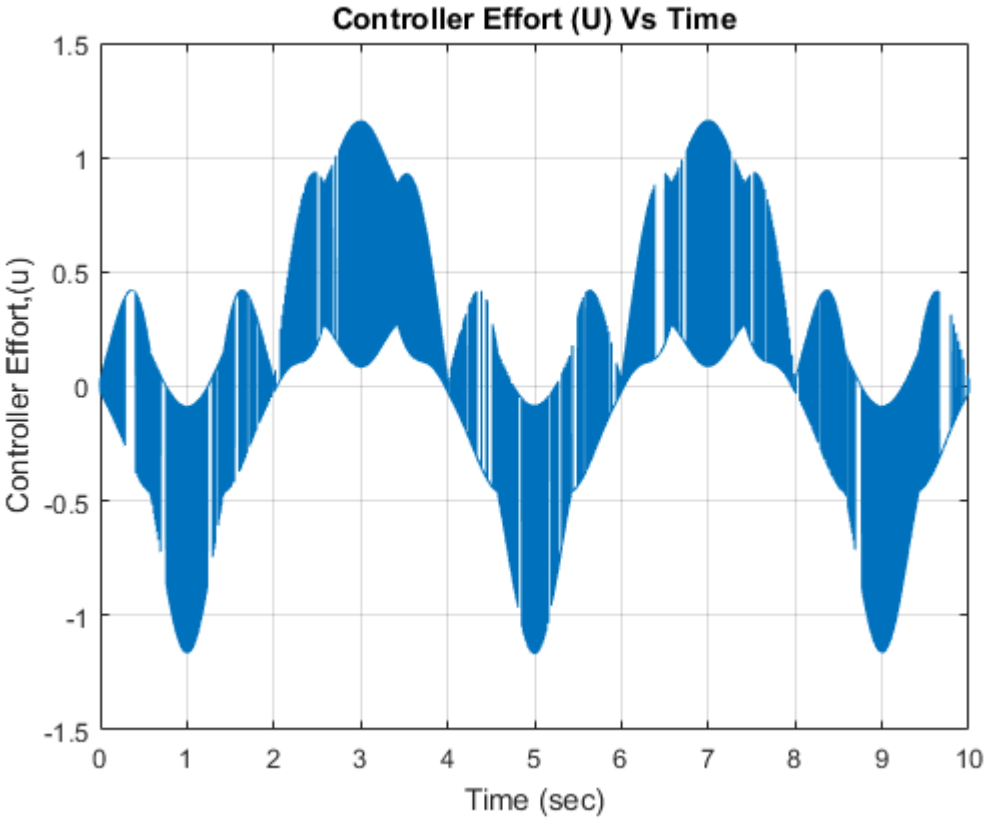

Figure 7: Controller Effort

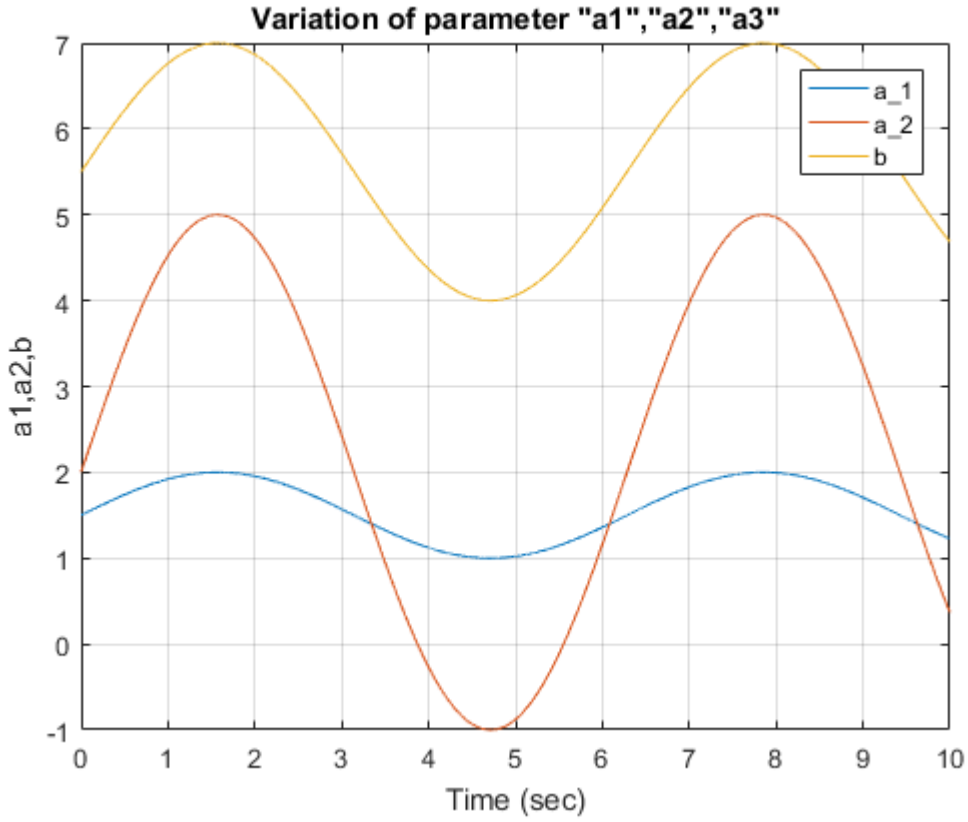

Figure 8: Variation of System Parameters, α_1, α_2, b

PLOTS: SATURATION FUNCTION

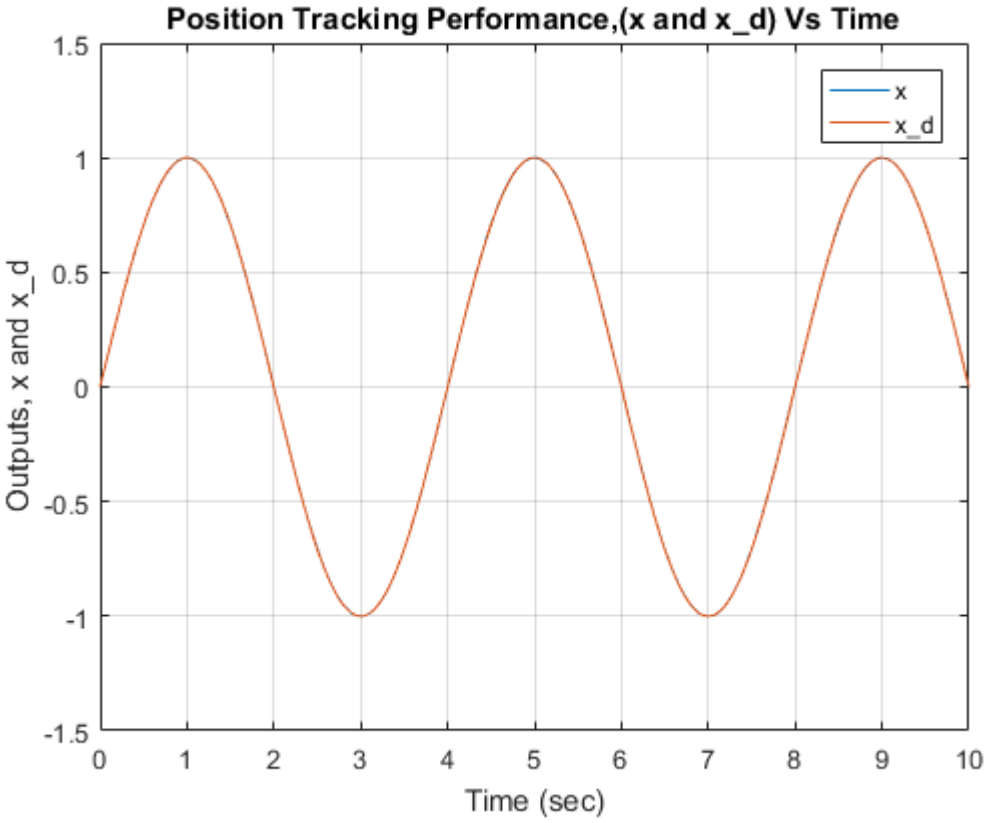

Figure 9: Position Tracking Performance

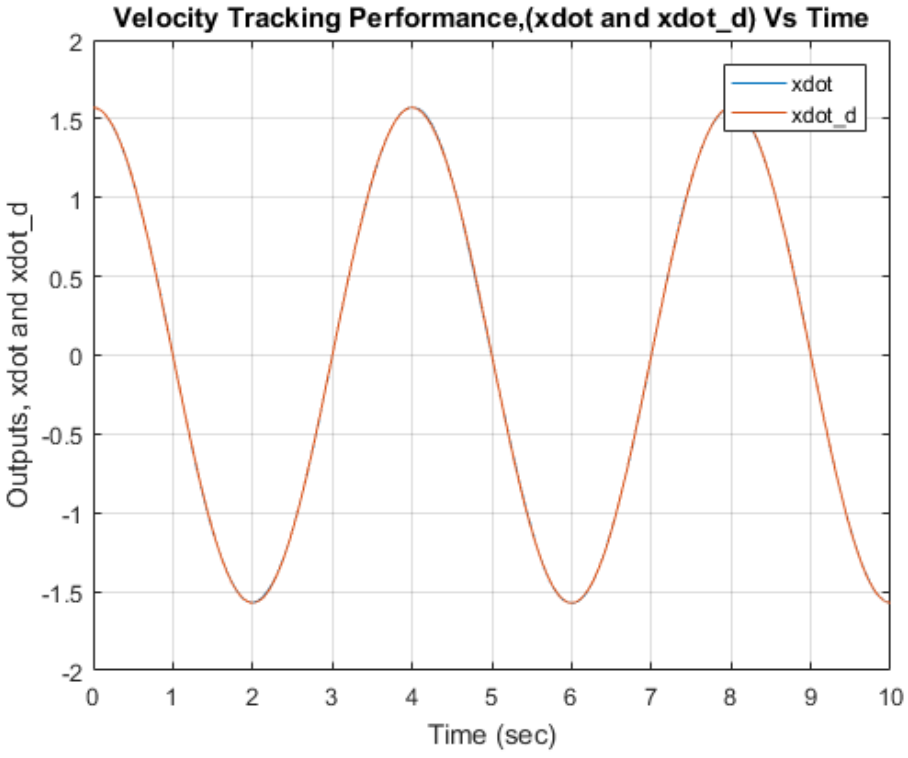

Figure 10: Velocity Tracking Performance

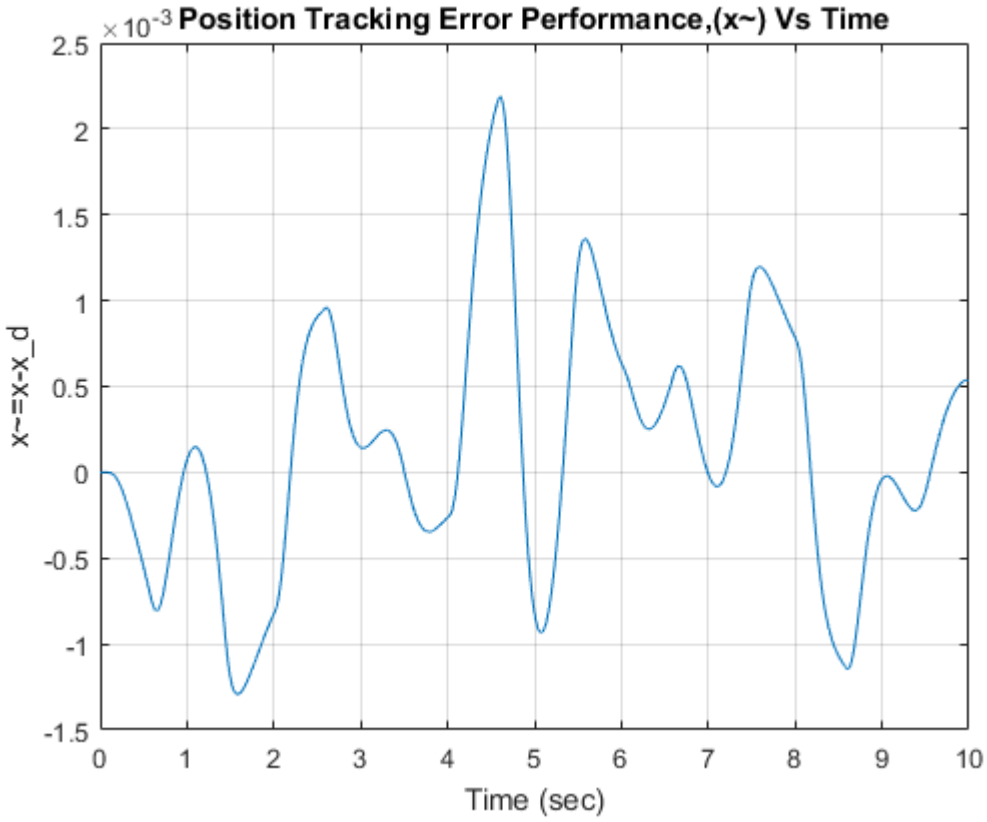

Figure 11: Position Tracking Error Performance

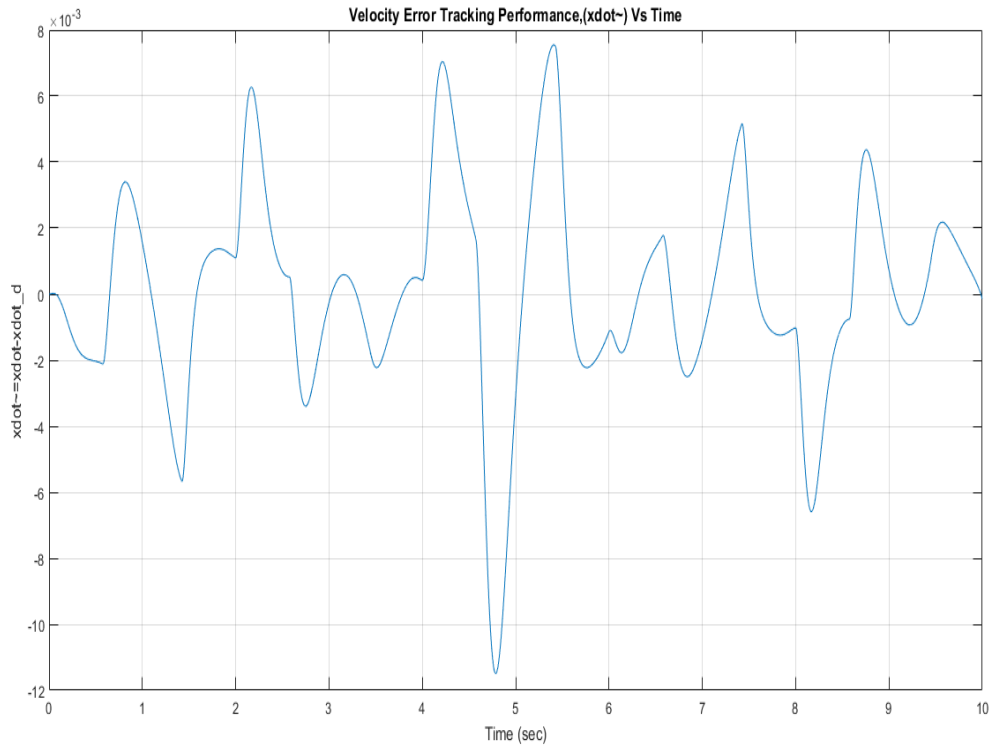

Figure 12: Velocity Tracking Error Performance

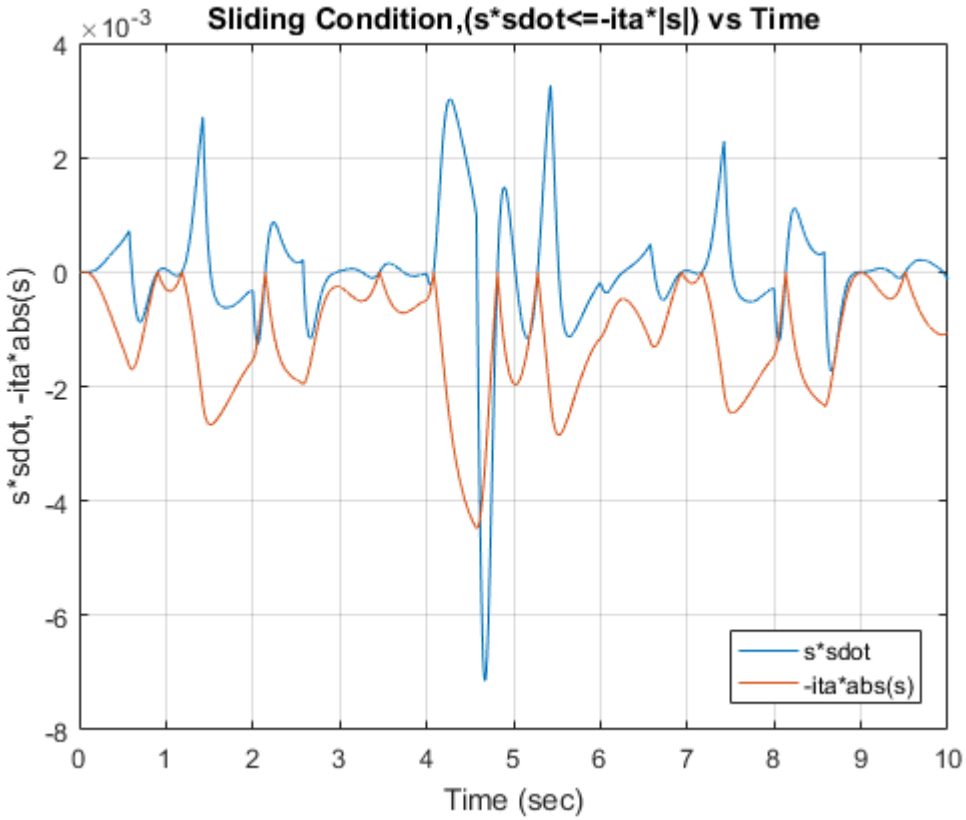

Figure 13: Sliding Condition

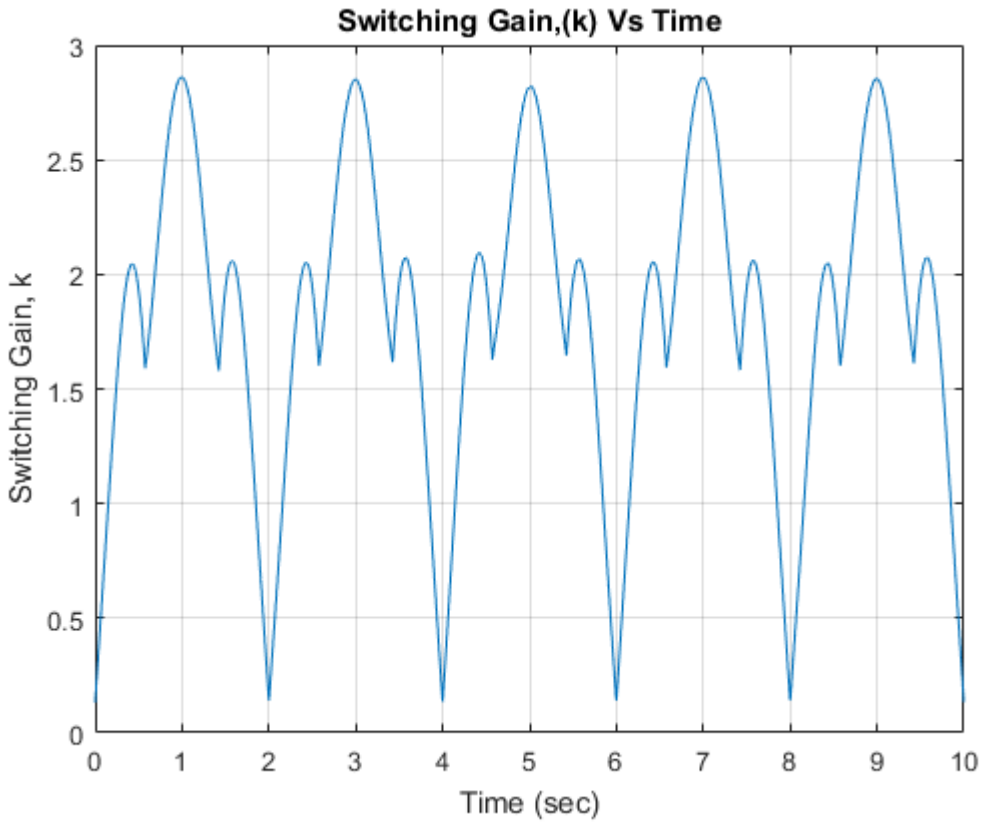

Figure 14: Switching Gain

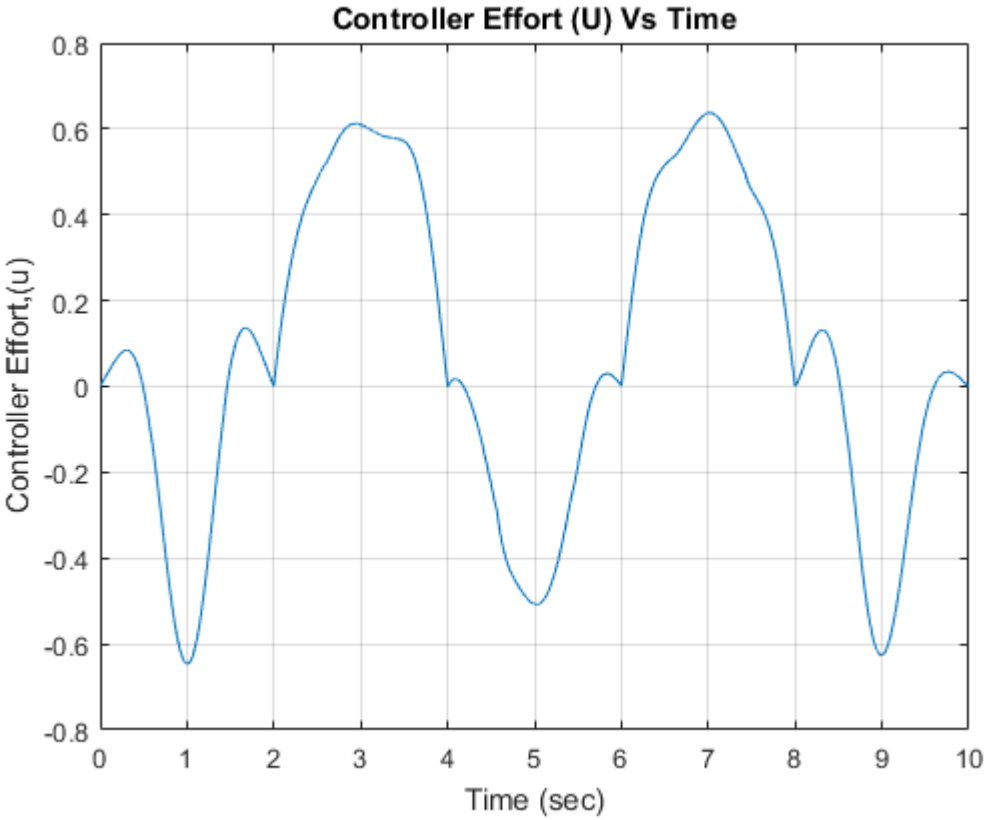

Figure 15: Controller Effort

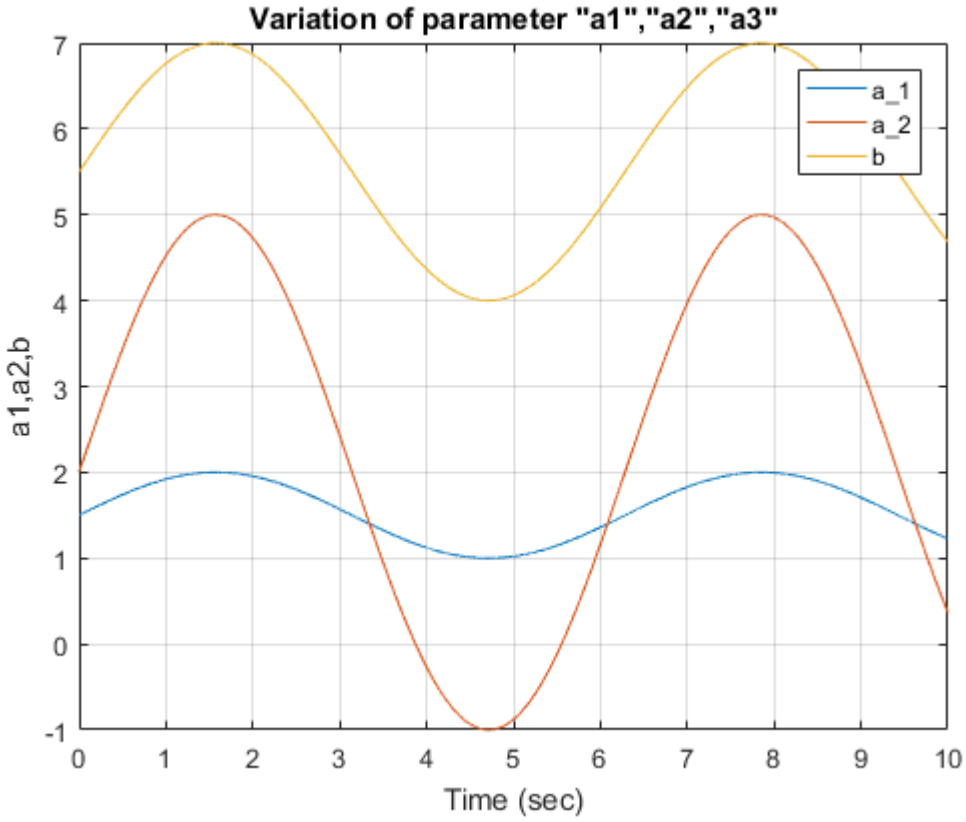

Figure 16: Variation of System Parameters, α_1, α_2, b

SIMULINK MODELS

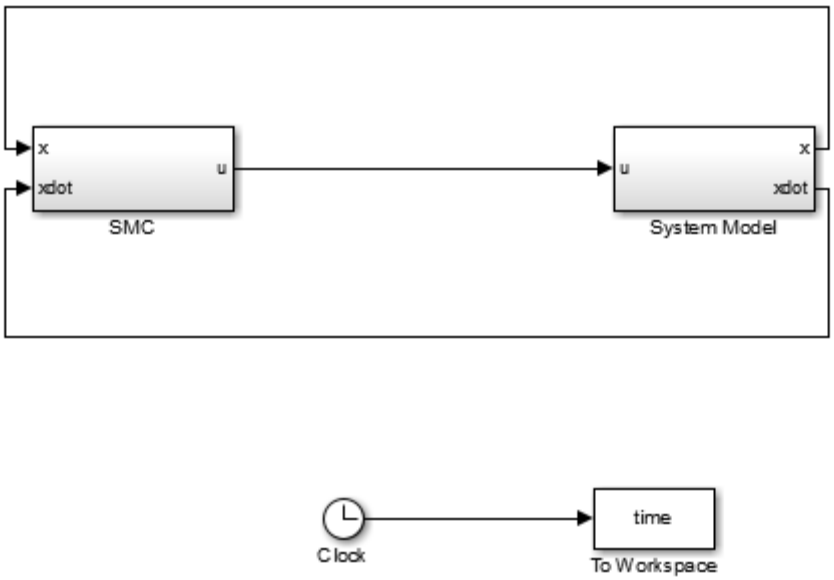

Figure 17: Main Block Diagram

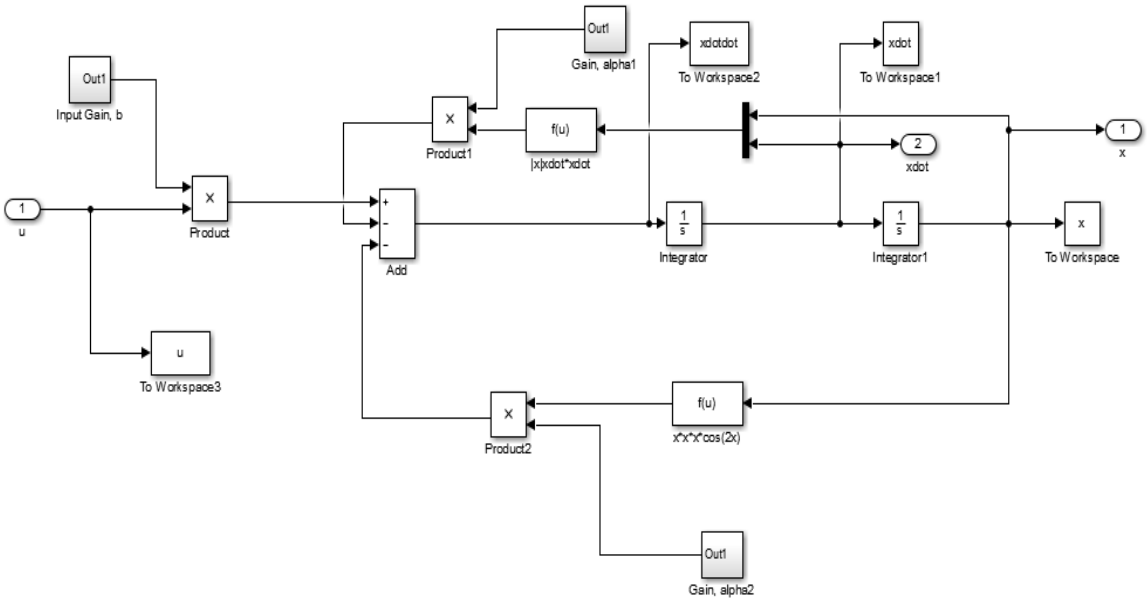

Figure 18: System Model

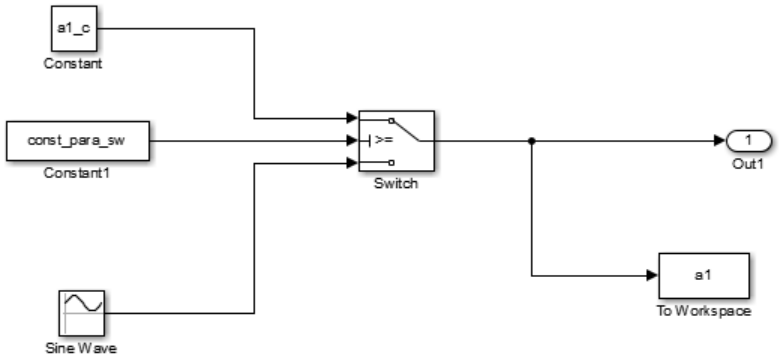

Figure 19: System Parameter α_1

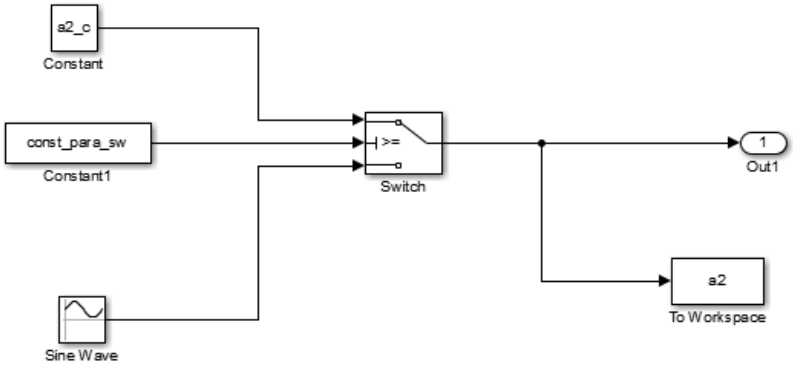

Figure 20: System Parameter α_2

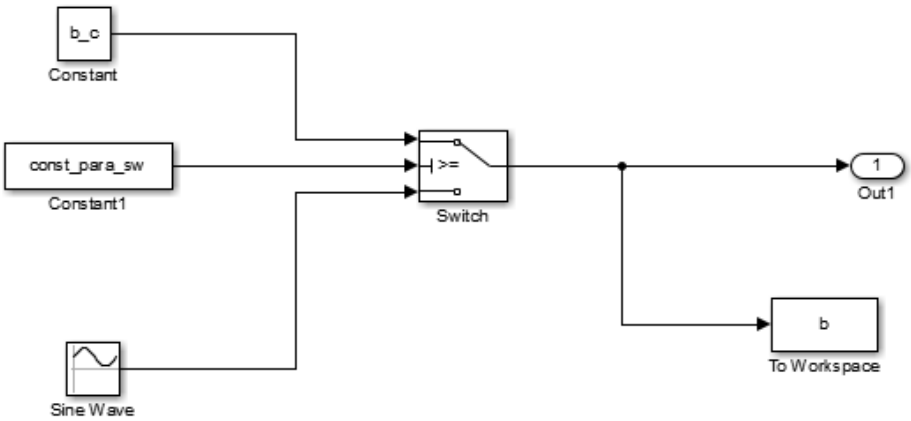

Figure 21: System Parameter b

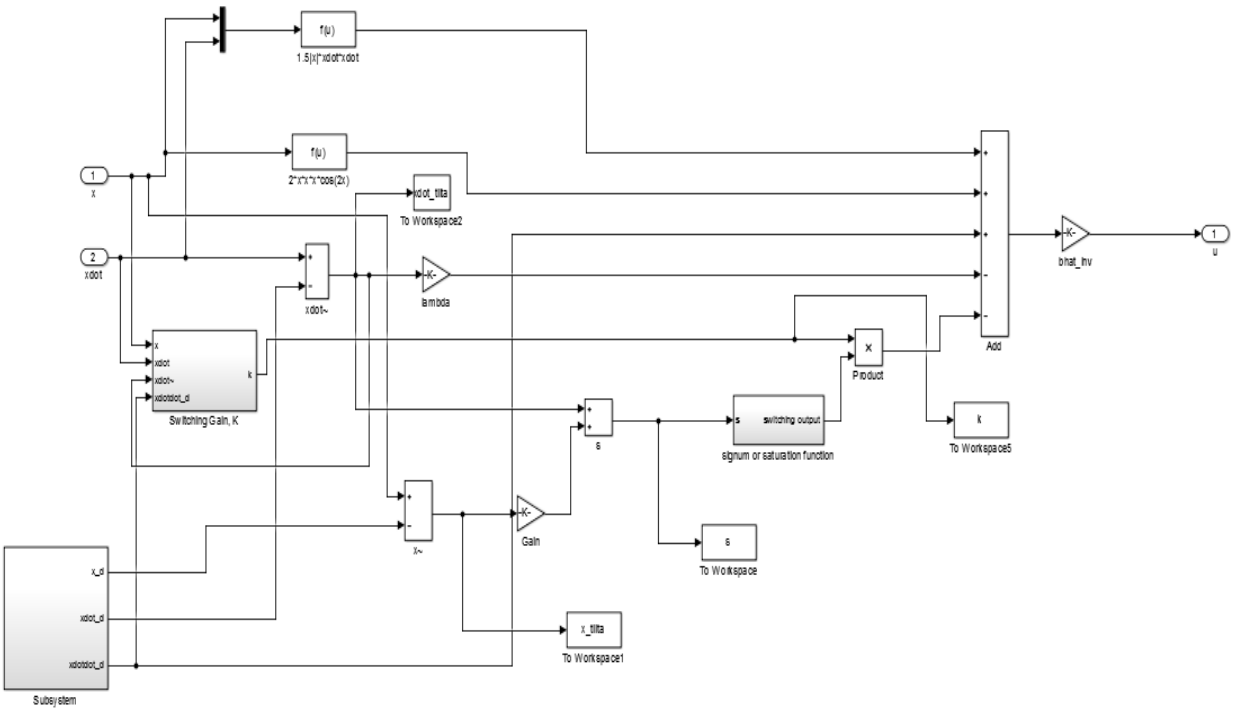

Figure 22: SMC Block

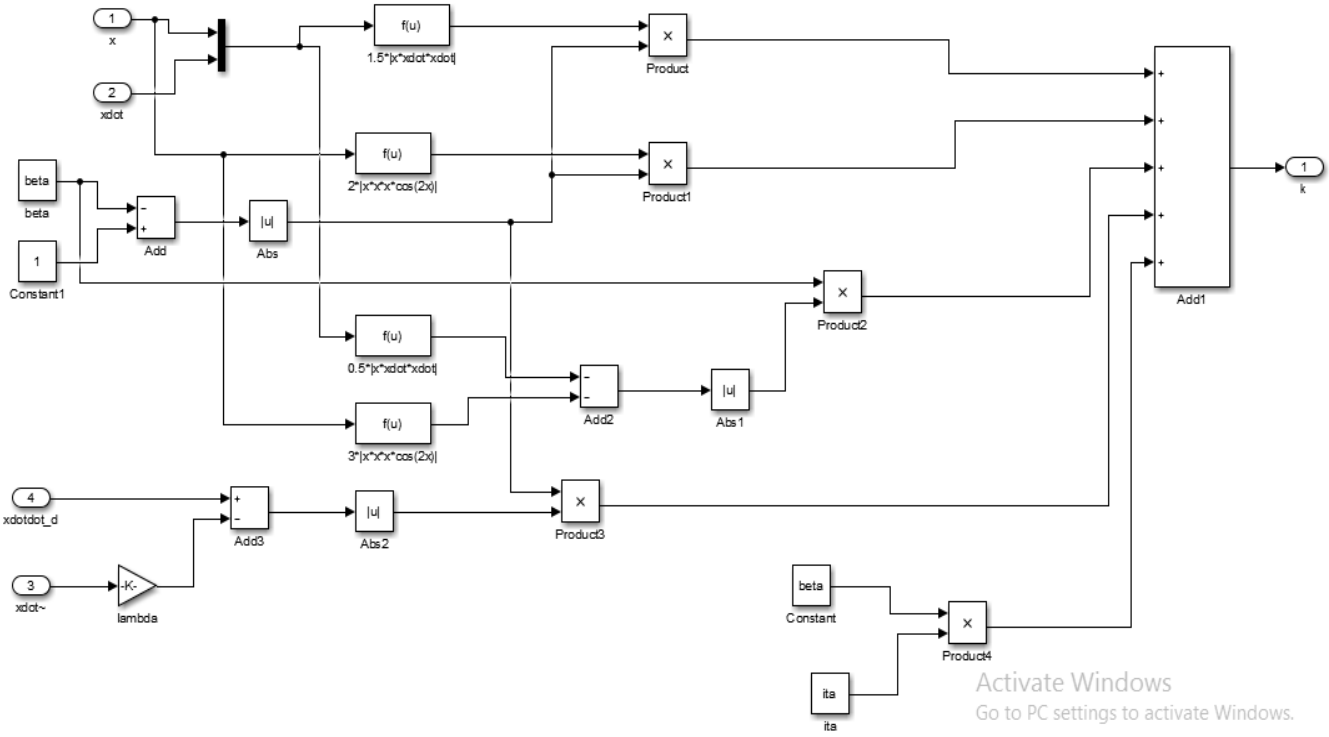

Figure 23: Switching Gain Block

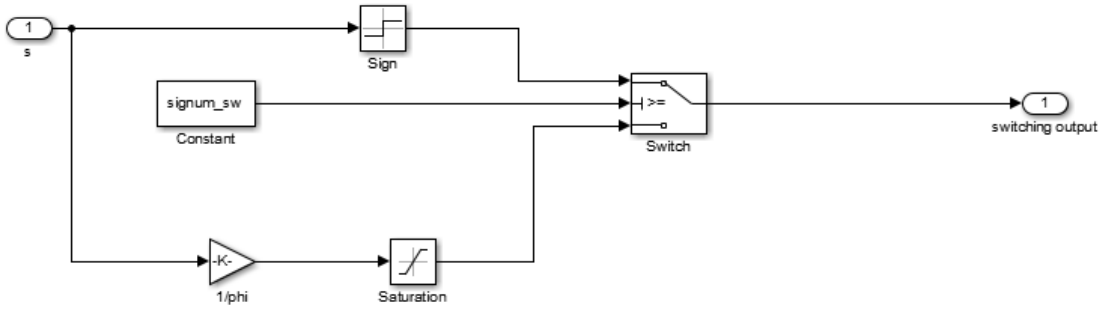

Figure 24: Signum / Saturation Function Switching Block

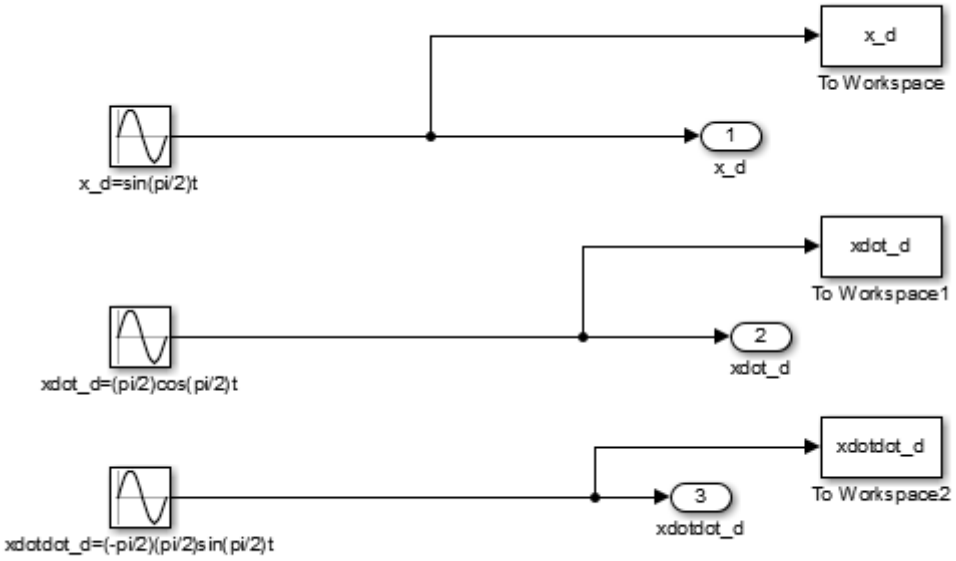

Figure 25: Desired Tracking Function Block

MATLAB CODE

```
                        %% Computer Project 4
                        %% Coded By K S Adarsh Raj
clear all,clc;

%% Switch between constant or varying system parameters
% const_parm_sw=input([1]:constant, [0]:vary);
const_para_sw=0;
if( isempty(const_para_sw) )
    const_para_sw=0;
end

%% Switch betwewn Signum or Saturation Function
% signum_sw=input([1]:signum, [0]:saturation);
signum_sw=1;
if( isempty(signum_sw) )
    signum_sw=1;
end

%% Define the Constant System Model Parameters
a1_c=1.5;
a2_c=2;
b_c=5.5;

%% Define Initial Condtions for x and xdot
x0=0;
xdot0=pi/2;

%% Define the SMC gains
lambda=20;
ita=0.1;
phi=0.1;

%% Define the upper and lower bounds for a1 and b
a1_upp=2;
a1_low=1;
b_upp=7;
b_low=4;

%% Define a1hat and bhat
a1hat=sqrt(a1_upp*a1_low);
bhat=sqrt(b_upp*b_low);

%% Define inverse of a1hat and bhat
a1hat_inv=1/a1hat;
bhat_inv=1/bhat;

%% Define alpha1 and beta
alpha1=sqrt(a1_upp/a1_low);
beta=sqrt(b_upp/b_low);

%% Run the simulation
sim('Project4');

%% Calculate sdot
sdot=(xdotdot-xdotdot_d)+lambda*(xdot-xdot_d);
```

```
%% Plot the Required Results
%Plot of x and x_d
figure(1),plot(time,x,time,x_d)
grid on
title('Position Tracking Performance,(x and x\_d) Vs Time')
xlabel('Time (sec)')
ylabel('Outputs, x and x\_d')
legend('x','x\_d')

%Plot of xdot and xdot_d
figure(2),plot(time,xdot,time,xdot_d)
grid on
title('Velocity Tracking Performance,(xdot and xdot\_d) Vs Time')
xlabel('Time (sec)')
ylabel('Outputs, xdot and xdot\_d')
legend('xdot','xdot\_d')

%Plot of x~
figure(3),plot(time,x_tilta)
grid on
title('Position Tracking Error Performance,(x~) Vs Time')
xlabel('Time (sec)')
ylabel('x~=x-x\_d')

%Plot of xdot~
figure(4),plot(time,xdot_tilta)
grid on
title('Velocity Error Tracking Performance,(xdot~) Vs Time')
xlabel('Time (sec)')
ylabel('xdot~=xdot-xdot\_d')

%Plot of Sliding Condition
figure(5),plot(time,s.*sdot,time,-ita*abs(s))
grid on
title('Sliding Condition,(s*sdot<=-ita*|s|) vs Time')
xlabel('Time (sec)')
ylabel('s*sdot, -ita*abs(s)')
legend('s*sdot','-ita*abs(s)','Location','SouthEast')

%Plot of Switching Gain
figure(6),plot(time,k)
grid on
title('Switching Gain,(k) Vs Time')
xlabel('Time (sec)')
ylabel('Switching Gain, k')

%Plot of Controller Effort
figure(7),plot(time,u)
grid on
title('Controller Effort (U) Vs Time')
xlabel('Time (sec)')
ylabel('Controller Effort,(u)')

%Plot of Varying Parameters
figure(8),plot(time,a1,time,a2,time,b)
grid on
title('Variation of parameter "a1","a2","a3"')
xlabel('Time (sec)')
ylabel('a1,a2,b')
legend('a\_1','a\_2','b')
```