|  |
| --- |
|  |
| Simulation 2  MAE 234 |
| **Layne Clemen** |
| **5/9/2013** |
|  |
|  |

|  |
| --- |
|  |

### System



Figure 1 - System Diagram

**State Equations**

|  |  |
| --- | --- |
|  | **(1)** |
|  | **(2)** |
|  | **(3)** |
|  | **(4)** |
|  | **(5)** |
|  | **(6)** |

The force equations above are only applicable to the reference vehicle. The real vehicle utilizes the Dugoff tire model to calculate the front and rear tire forces during simulation. Those equations have been omitted but re readily available.

### Gains

The gains, KI and KP, were found using the PID tool in MATLAB. Using the gains from calculated with the tool, it was possible to establish relationships for KI and KP as functions of the forward speed U. the equations are below

|  |  |
| --- | --- |
|  | **(7)** |
|  | **(8)** |

These gains gave good responses as seen below. These were calculated with KU = 1.1 for the reference car. The Controller appears to be somewhat robust for changes in KU.

### Outputs and Analysis

The yaw rate tracked very well across most of the operating conditions for both KU = 0.8 and KU = 1.2 for the real car, except for a δ = 10 degrees. Also for δ = 8 degrees and a KU = 0.8, the yaw rate did not track. See figures 2 and 3 for examples.

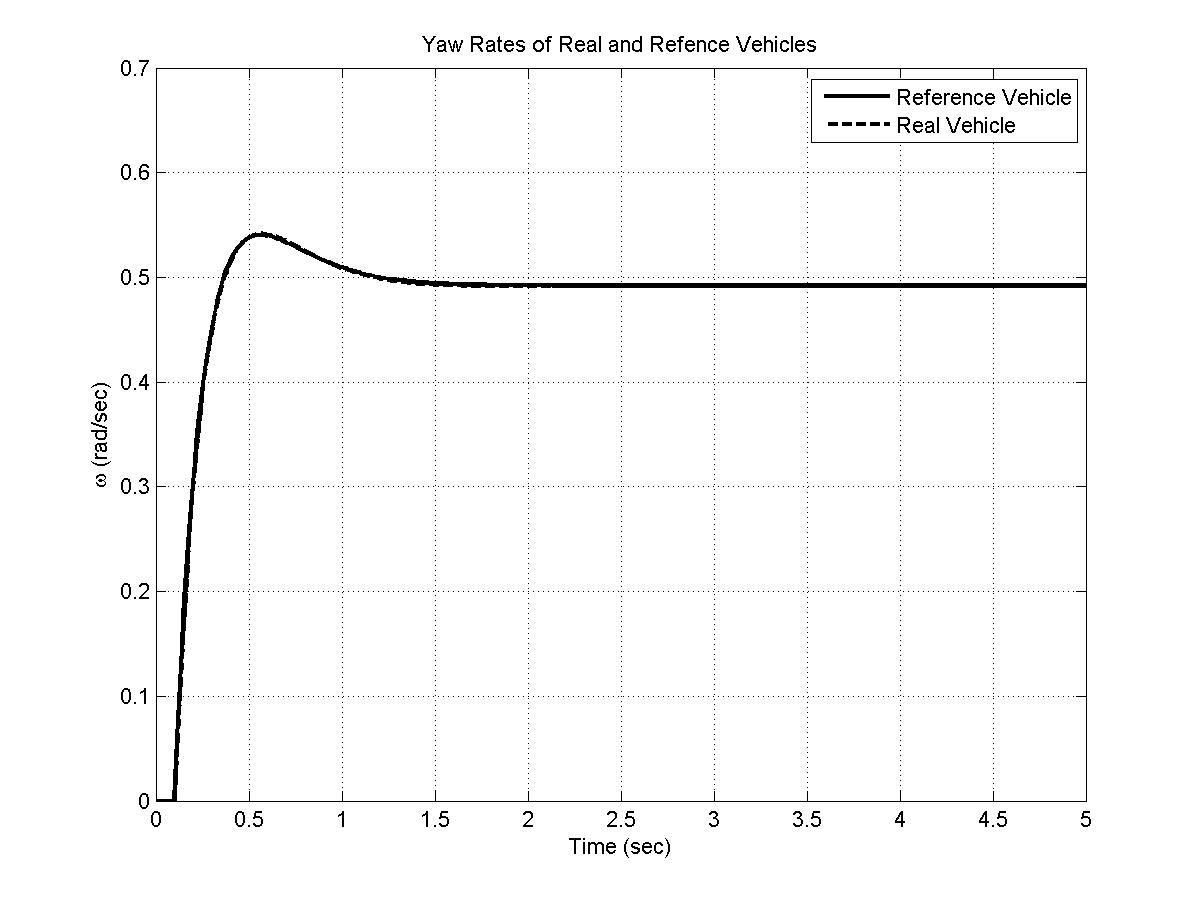


Figure 2 – Stable yaw rate responses for U = 60 mph, δ = 6 degrees, Ku = 1.2

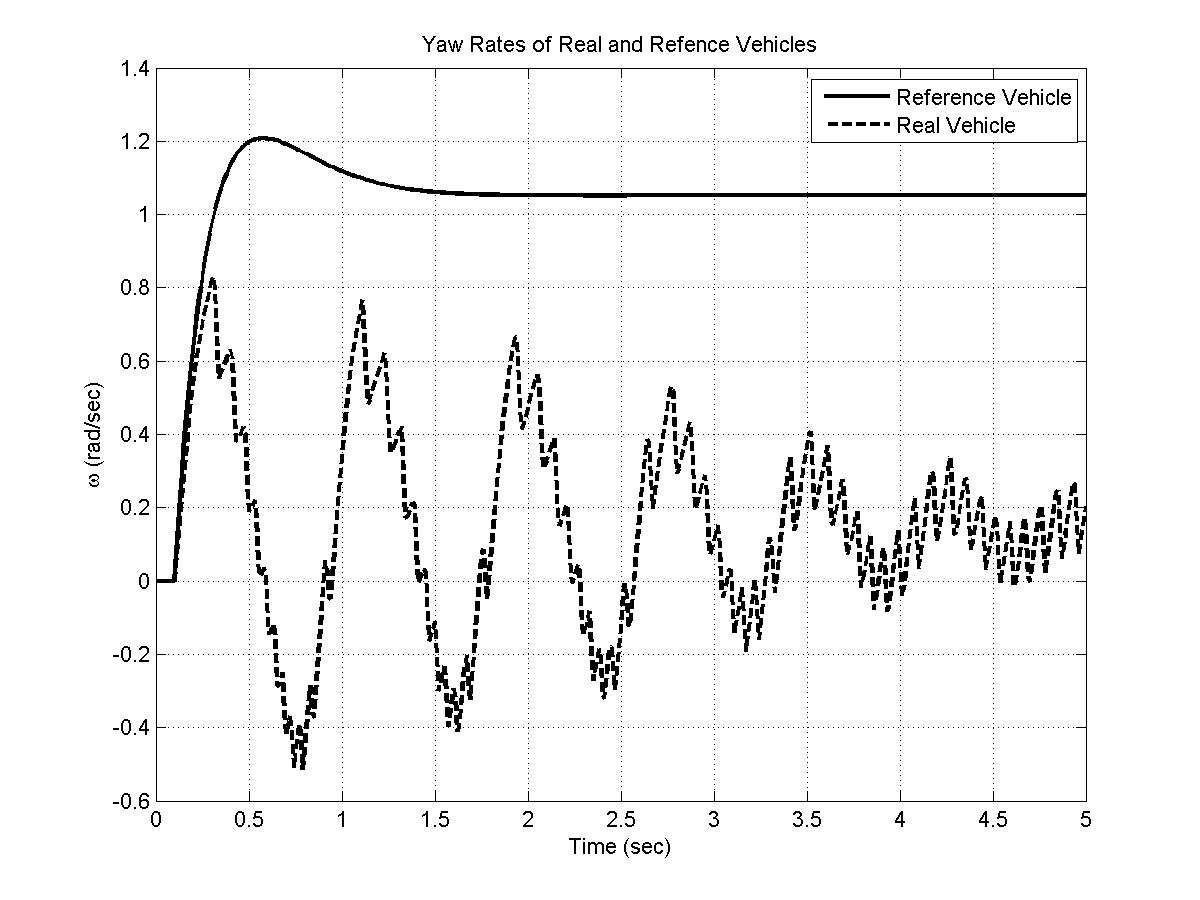


Figure 3 - Nontracking yaw rate with U = 70 mph, δ = 8 degrees, Ku = 1.2

Not much difference was observed in the yaw rate responses for KU = 0.8 versus the KU = 1.2.The steering angles were mostly believable up to input magnitudes above 6 degrees. After this the commanded steer angles either had maximum amplitudes of hundreds of degrees or they went unstable. Examples of these can be seen in figures 4 and 5. Something that was observed was that in the simulation for U = 60 mph, δ = 6 degrees, and KU = 1.2. In the middle of the simulation the yaw rate diverged (see figure 6) and then came back to track the yaw rate again. It was hypothesized that this might be due to the gain schedule. When other gains were tried, such as KI = 1 and KP = 70, this fixed the problem for these simulation conditions but the other simulations did not display smooth command steering angles like those that were seen with the gain scheduled simulations. If this study were moving forward, further investigation in the gain schedule and optimizing it would be recommended.

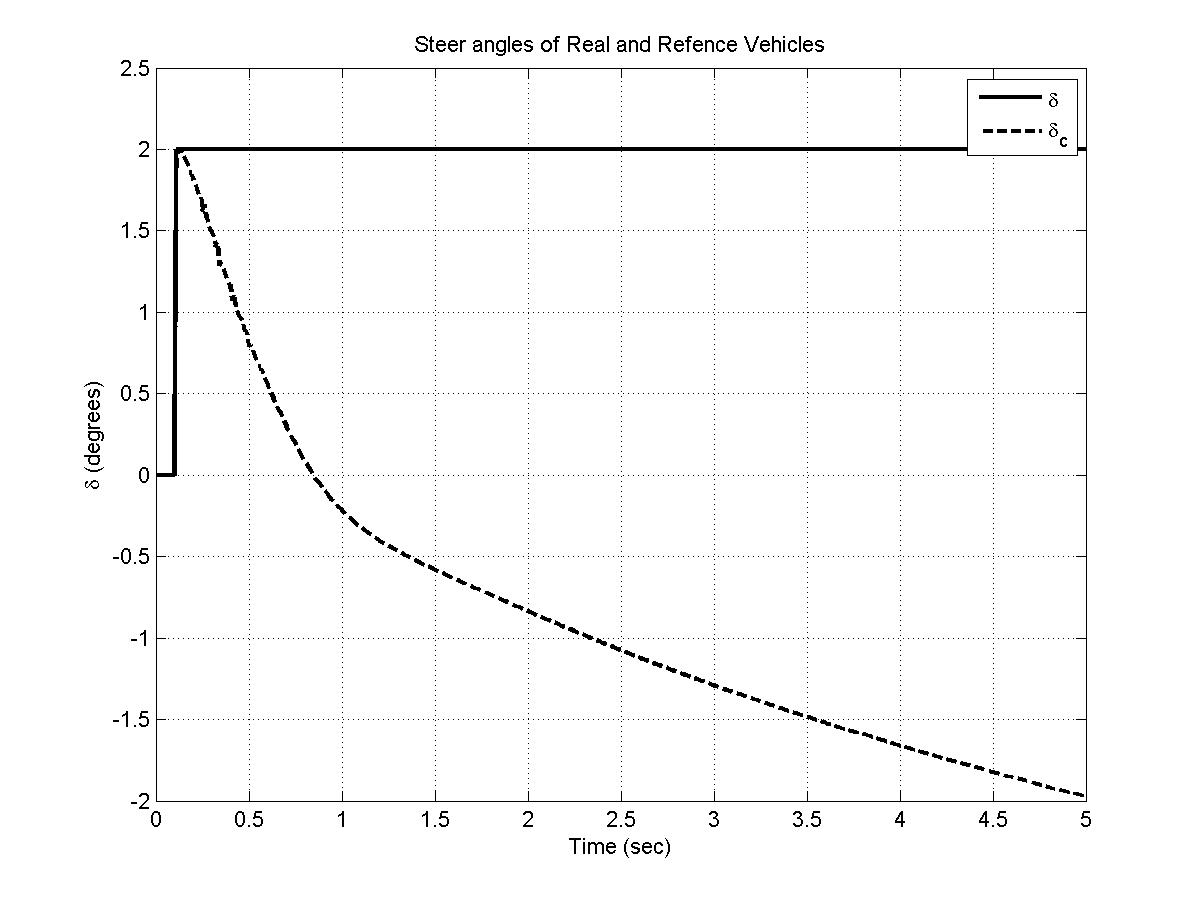


Figure 4 - Stable steer for step input with U = 70 mph, δ = 2 degrees, Ku = 0.8

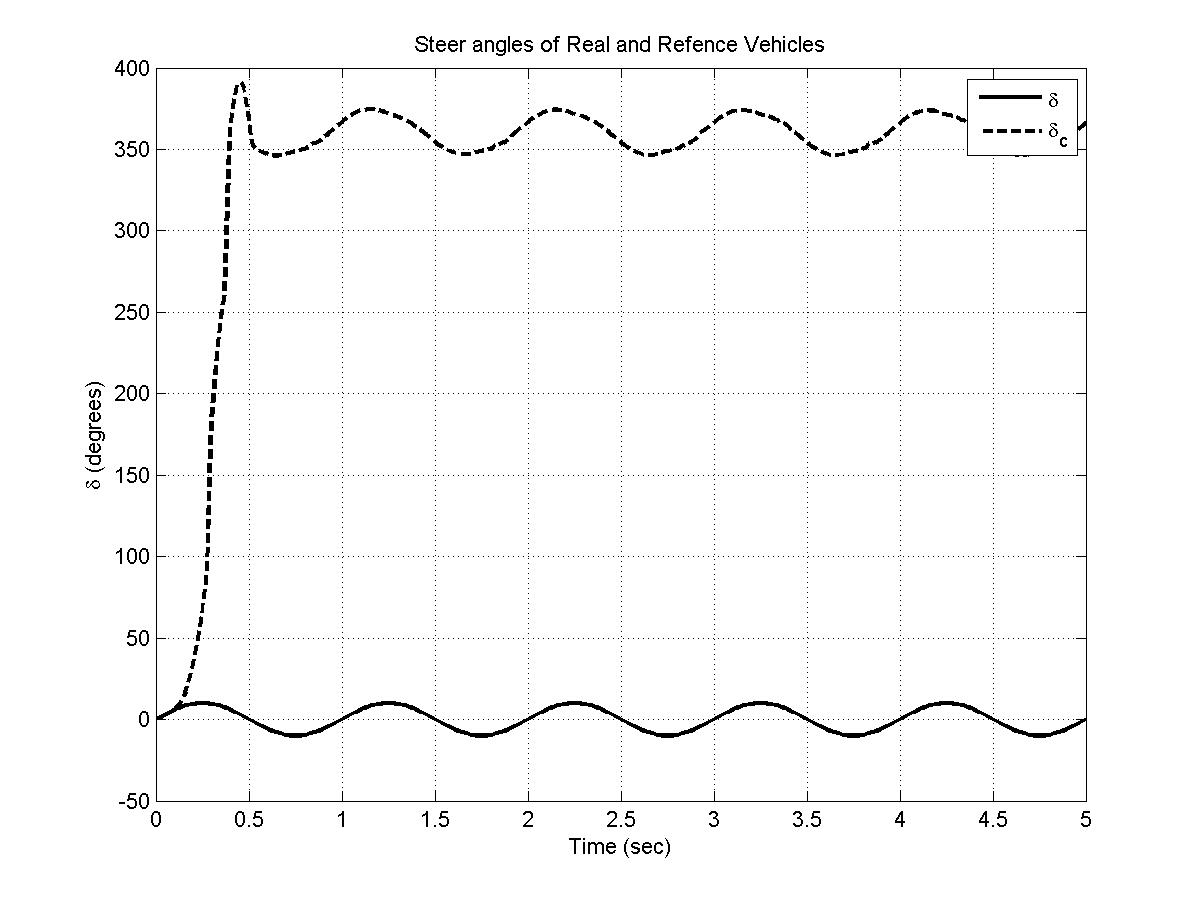


Figure 5 - Unstable steer angles for sinusoidal steer input with with U = 70 mph, δ = 10 degrees, Ku = 0.8

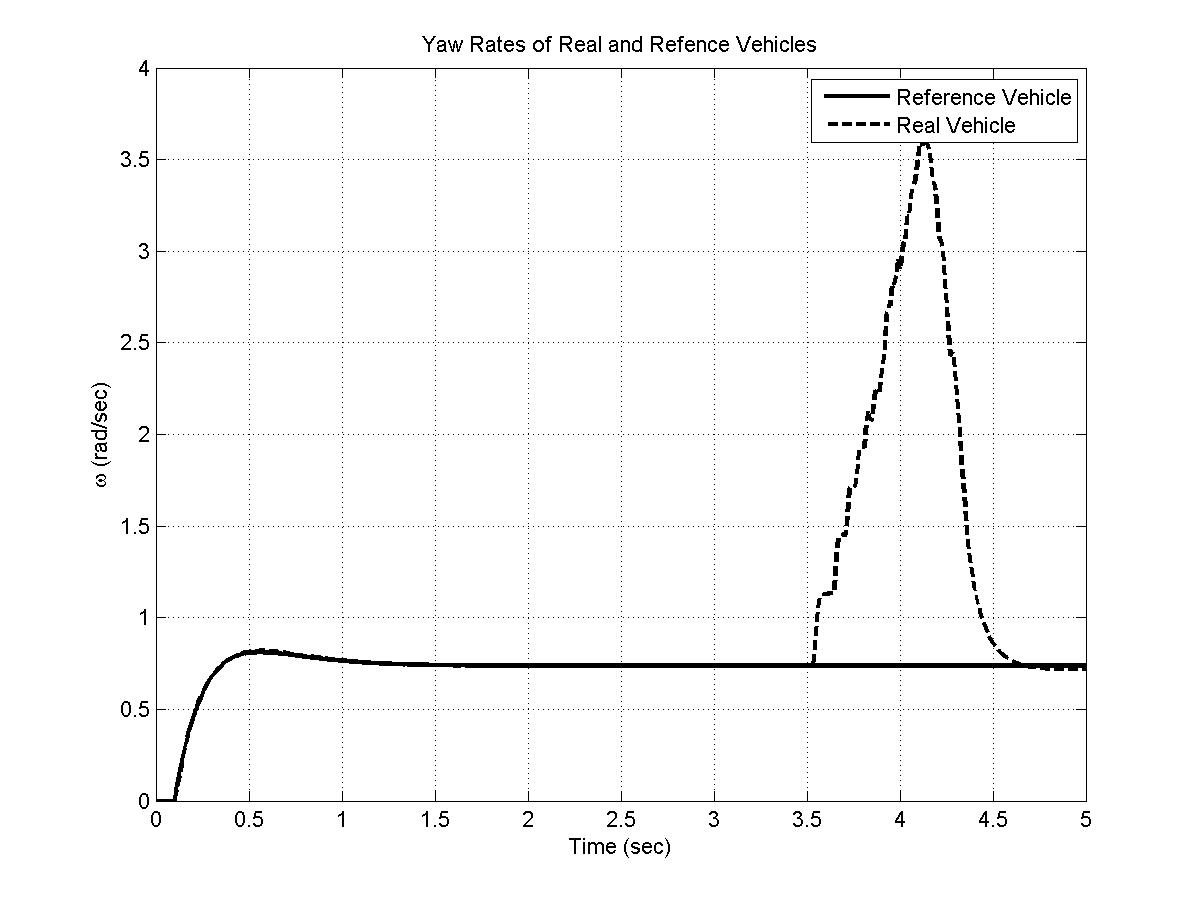


Figure 6 - Strange yaw rate behavior with U = 60 mph, δ = 6 degrees, Ku = 1.2

The trajectories did not match up for most of the simulations. At lower velocities and smaller delta inputs, the real vehicle was able to track the trajectory of the reference vehicle. This can be explained by the use of the Dugoff tire model that is used on the real car. Very quickly with the simulations the tire forces saturate and tracking of the trajectory is not possible because the vehicle is not able to generate the required forces to match the lateral acceleration of the reference vehicle. See figures 7, 8, and 9 for examples of this. From figure 8 the tire saturation due to the Dugoff model is apparent. The lateral acceleration differences in figure 7 lead to the conclusion that the y-coordinate of the vehicle will not track the reference vehicle which is what is observed in figure 9.

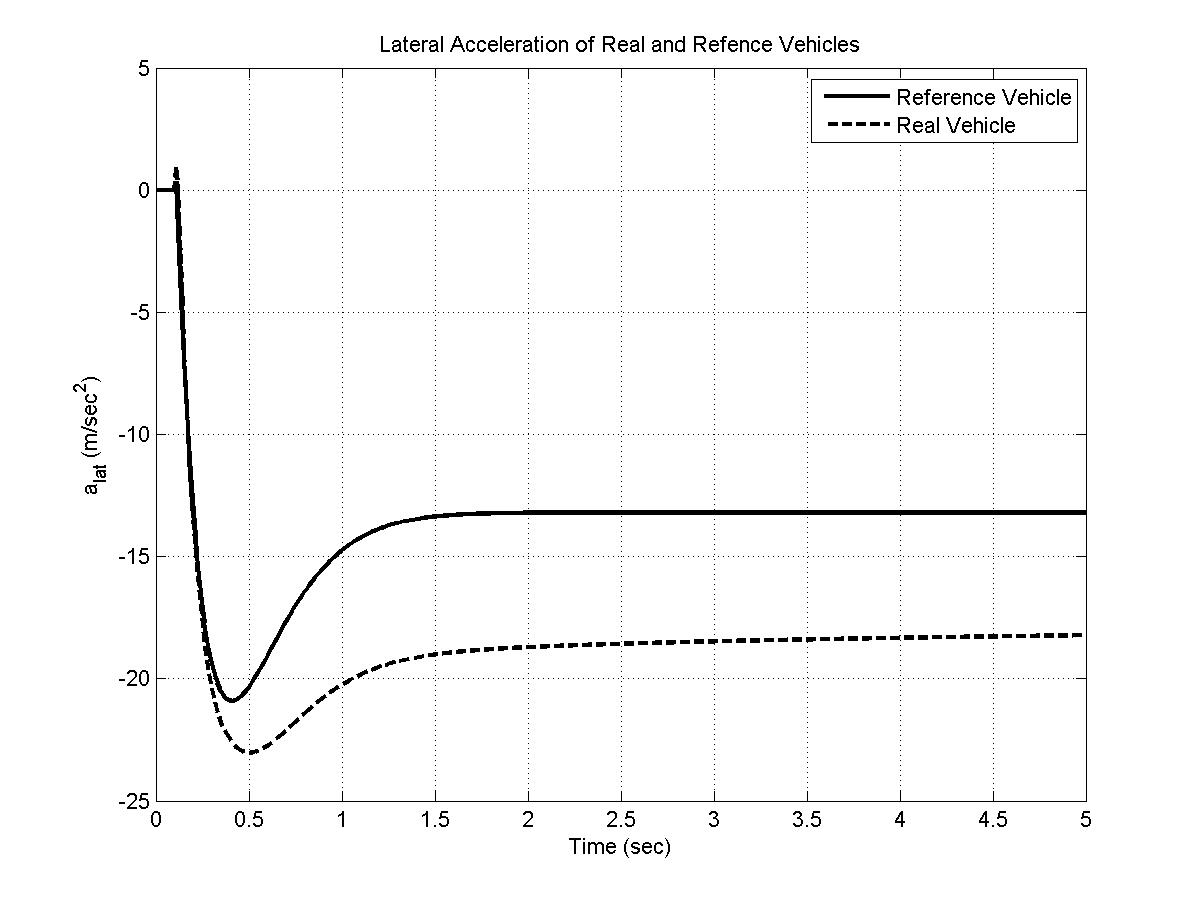


Figure - Lateral Acceleration for U = 60 mph, δ = 4 degrees, Ku = 0.8

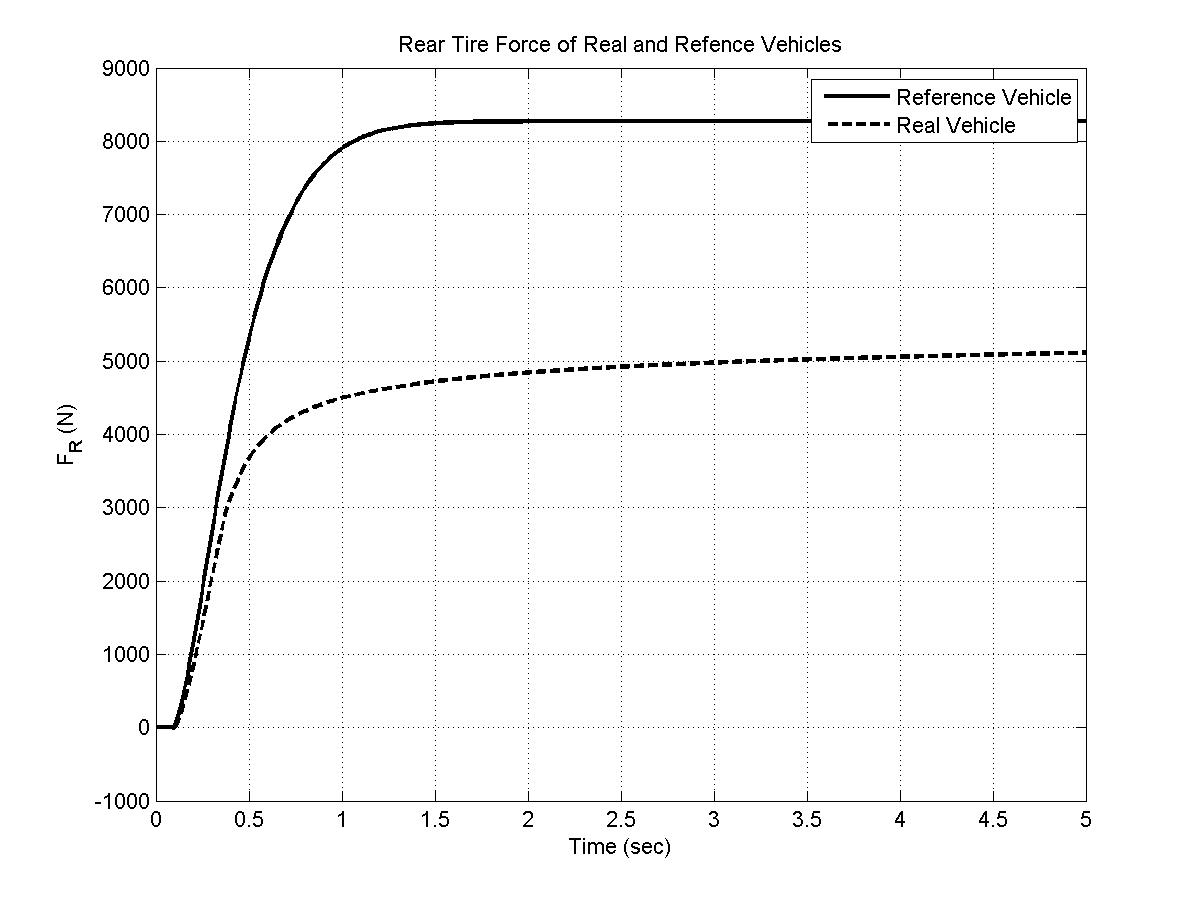


Figure - Rear Tire Forces U = 60 mph, δ = 4 degrees, Ku = 0.8

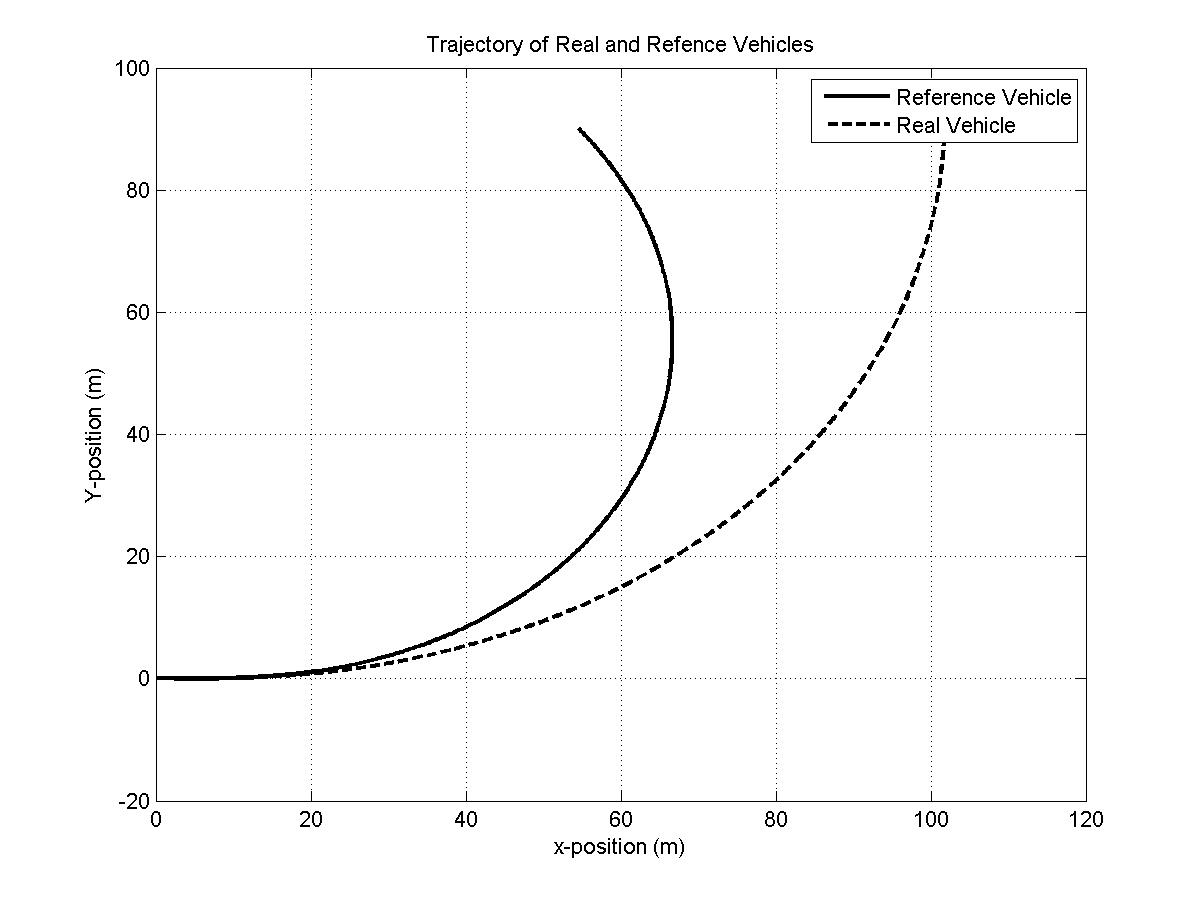


Figure - Trajectory for U = 60 mph, δ = 4 degrees, Ku = 0.8

### Appendix A – MATLAB Code

master.m:

%% Simulation 2 - Layne Clemen

%% MAE 234, Spring Term 2013 - Dr. Margolis

% Bicycle

%% Clean up

clc; close all; clear all;

%% Parameters

global m b a Cf Jy U Ku\_MR Cr\_MR mu Ku\_R Cr\_R sf sr Nf Nr ti d0

% Parameters common to real and reference vehicle

m = 3000/2.2; % kg, Total mass

L = 2.84; % m, Wheelbase

rab = 0.85; % Ratio of a to b

b = L/(1+rab); % m, Length from cg to rear axle

a = L-b; % m, length from cg to front axle

Jy = 0.4\*m\*a\*b; % kg-m^2, Yaw moment of inertia

Cf = 50000; % N/rad, Front cornering coefficient in the linear range

U = 50\*0.447; %Forward velocity - m/s

g = 9.81; % m/s^2, Gravity

d0 = 2;

% Reference vehicle parameters

Ku\_MR = 1.1; % Model reference understeer coefficient

Cr\_MR = Cf\*rab\*Ku\_MR; % N/rad, rear cornering coefficient

% Real vehicle parameters

mu = 0.85; % Coefficeint of kinetic friction

Ku\_R = 0.8; % Real vehicle understeer coefficient

Cr\_R = Cf\*rab\*Ku\_R; % N/rad, Real vehicle rear cornering coefficient

sf = 0; % Front longitudinal slip ratio

sr = 0; % Rear longitudinal slip ratio

Nf = b\*m\*g/(a+b); % N, Front normal force

Nr = a\*m\*g/(a+b); % N, Rear normal Force

%% Time vector

tspan = 0:0.01:5;

ti = 0.1;

%% Initial Conditions - Reference Car

% [V0, theta0, omega0, x0, y0]

V0 = 0; % m/s, inital lateral velocity

theta0 = 0; % rad, initial yaw angle

omega0 = 0; % rad/s, initial yaw rate

x0 = 0; % m, initial x coordinate

y0 = 0; % m, initial y coordinate

%% Initial Conditions - Real Car

% [V0, theta0, omega0, x0, y0]

V0\_R = 0; % m/s, inital lateral velocity

theta0\_R = 0; % rad, initial yaw angle

omega0\_R = 0; % rad/s, initial yaw rate

x0\_R = 0; % m, initial x coordinate

y0\_R = 0; % m, initial y coordinate

erri0 = 0; %rad, initial yaw rate error

initial = [V0 theta0 omega0 x0 y0 V0\_R theta0\_R omega0\_R x0\_R y0\_R erri0];

%% Simulation

[t,s] = ode45(@eqns,tspan,initial);

for i = 1:length(t)

[ds(i,:) ext(i,:)] = eqns(t(i), s(i,:));

end

%% Plots

traj = ['trajec.' num2str(U/0.447) '.' num2str(d0) '.step.jpg']

RF = ['RearForces.' num2str(U/0.447) '.' num2str(d0) '.step.jpg']

a\_lat = ['a\_lat.' num2str(U/0.447) '.' num2str(d0) '.step.jpg']

steer = ['SteerAngle.' num2str(U/0.447) '.' num2str(d0) '.step.jpg']

yaw = ['YawRate.' num2str(U/0.447) '.' num2str(d0) '.step.jpg']

figure('Name','Trajectory','NumberTitle','off','Color','white')

plot(s(:,4),s(:,5),'-k',s(:,9),s(:,10),'--k','LineWidth',2), grid on

title('Trajectory of Real and Refence Vehicles')

legend('Reference Vehicle','Real Vehicle')

ylabel('Y-position (m)')

xlabel('x-position (m)')

saveas(gcf,traj)

figure('Name','Rear Forces','NumberTitle','off','Color','white')

plot(t,ext(:,2),'-k',t,ext(:,7),'--k','LineWidth',2), grid on

title('Rear Tire Force of Real and Refence Vehicles')

legend('Reference Vehicle','Real Vehicle')

ylabel('F\_{R} (N)')

xlabel('Time (sec)')

saveas(gcf,RF)

figure('Name','Steer Angle Inputs','NumberTitle','off','Color','white')

plot(t,ext(:,5)\*180/pi,'-k',t,ext(:,10)\*180/pi,'--k','LineWidth',2), grid on

title('Steer angles of Real and Refence Vehicles')

legend('\delta','\delta\_c')

ylabel('\delta (degrees)')

xlabel('Time (sec)')

saveas(gcf,steer)

figure('Name','Yaw Rate','NumberTitle','off','Color','white')

plot(t,s(:,3),'-k',t,s(:,8),'--k','LineWidth',2), grid on

title('Yaw Rates of Real and Refence Vehicles')

legend('Reference Vehicle','Real Vehicle')

ylabel('\omega (rad/sec)')

xlabel('Time (sec)')

saveas(gcf,yaw)

figure('Name','Lateral Acceleration','NumberTitle','off','Color','white')

plot(t,(ds(:,1)-U\*s(:,3)),'-k',t,(ds(:,6)-U\*s(:,8)),'--k','LineWidth',2), grid on

title('Lateral Acceleration of Real and Refence Vehicles')

legend('Reference Vehicle','Real Vehicle')

ylabel('a\_{lat} (m/sec^2)')

xlabel('Time (sec)')

saveas(gcf,a\_lat)

eqns.m:

%% Equations file for simulation 2 - Planar handling model.

function [ds, ext] = eqns(t,s)

%% Parameters

global m b a Cf Jy U Ku\_MR Cr\_MR mu Ku\_R Cr\_R sf sr Nf Nr ti d0

%% Variable initialization - omega = theta\_dot

% Reference car

V = s(1);

theta = s(2);

omega = s(3);

x = s(4);

y = s(5);

%Real car

V\_R = s(6);

theta\_R = s(7);

omega\_R = s(8);

x\_R = s(9);

y\_R = s(10);

erri = s(11);

ds = zeros(11,1);

ext = zeros(10,1);

%% Inputs

% Delta

if t > ti

delta = d0\*pi/180;

else

delta = 0;

end

%

% delta = d0\*pi/180\*sin(2\*pi\*t);

%% Gain schedule for Ki and Kp

% This was found using the MATLAB PID tuning tool. The final equations were

% found by tuning the gains for a settling time of magnitude 10^-2 and then

% assuming that the gains for change from the found Ts to a Ts of 0.02 were

% proportional. These were then curve fit in Excel.

Ki = 0.5\*1903\*U^(-0.916);

Kp = 0.5\*(0.0556\*U + 6.0513);

%% Calculation of side force - Reference Car

% alpha = wheel slip angle and F = Ca\*alpha

alpha\_f = delta - (V + a\*omega)/U;

alpha\_r = (b\*omega-V)/U;

Ff = Cf\*alpha\_f;

Fr = Cr\_MR\*alpha\_r;

%% Calculation of side force - Real Car

% alpha = wheel slip angle and F = Ca\*alpha

err = omega-omega\_R;

delta\_c = Kp\*err + Ki\*erri;

alpha\_f\_R = delta\_c - (V\_R + a\*omega\_R)/U;

alpha\_r\_R = (b\*omega\_R-V\_R)/U;

%For the front

F\_xd\_f=Cf\*sf/(1-sf);

F\_yd\_f=Cf\*tan(alpha\_f\_R)/(1-sf);

Q\_f=sqrt( (F\_xd\_f/Nf)^2+(F\_yd\_f/Nf)^2 );

if Q\_f==0;

Q\_f=.0001; %Makes sure there is no divide by zero

end;

lambda\_f=mu/2/Q\_f;

if lambda\_f >=1;

Fx\_f=F\_xd\_f; Fy\_f=F\_yd\_f;

else

Fx\_f=F\_xd\_f\*2\*lambda\_f\*(1-lambda\_f/2);

Fy\_f=F\_yd\_f\*2\*lambda\_f\*(1-lambda\_f/2);

end;

Ff\_R=Fy\_f;

%For the rear

F\_xd\_r=Cr\_R\*sr/(1-sr);

F\_yd\_r=Cr\_R\*tan(alpha\_r\_R)/(1-sr);

Q\_r=sqrt((F\_xd\_r/Nr)^2+(F\_yd\_r/Nr)^2 );

if Q\_r == 0;

Q\_r=.0001; %Makes sure there is no divide by zero

end;

lambda\_r=mu/2/Q\_r;

if lambda\_r >=1;

Fx\_r=F\_xd\_r; Fy\_r=F\_yd\_r;

else

Fx\_r=F\_xd\_r\*2\*lambda\_r\*(1-lambda\_r/2);

Fy\_r=F\_yd\_r\*2\*lambda\_r\*(1-lambda\_r/2);

end;

Fr\_R=Fy\_r;

%% Equations of Motion - Reference Car

V\_dot = Ff/m + Fr/m - U\*omega;

theta\_dot = omega;

omega\_dot = a\*Ff/Jy - b\*Fr/Jy;

x\_dot = U\*cos(theta) - V\*sin(theta);

y\_dot = U\*sin(theta) + V\*cos(theta);

%% Equations of Motion - Real Car

V\_dot\_R = Ff\_R/m + Fr\_R/m - U\*omega\_R;

theta\_dot\_R = omega\_R;

omega\_dot\_R = a\*Ff\_R/Jy - b\*Fr\_R/Jy;

x\_dot\_R = U\*cos(theta\_R) - V\_R\*sin(theta\_R);

y\_dot\_R = U\*sin(theta\_R) + V\_R\*cos(theta\_R);

erri\_dot = err;

%% Extra outputs

ext(1) = Ff;

ext(2) = Fr;

ext(3) = alpha\_f;

ext(4) = alpha\_r;

ext(5) = delta;

ext(6) = Ff\_R;

ext(7) = Fr\_R;

ext(8) = alpha\_f\_R;

ext(9) = alpha\_r\_R;

ext(10) = delta\_c;

ds = [V\_dot; theta\_dot; omega\_dot; x\_dot; y\_dot;....

V\_dot\_R; theta\_dot\_R; omega\_dot\_R; x\_dot\_R; y\_dot\_R; erri\_dot];