```
function [displacement, velocity, acceleration] = Ride_Comfort_Analysis(car, stiffness,↙
damping, road, vel, acc)

m   = car.mass;
m1  = car.front_unsprung_mass;
m2  = car.rear_unsprung_mass;
Iy  = car.Lateral_MOI;
a1  = car.CG_2_Front;
a2  = car.CG_2_Rear;
k1  = stiffness.front_strut;
k2  = stiffness.rear_strut;
kt1 = stiffness.tire_front;
kt2 = stiffness.tire_rear;
c1  = damping.strut_front;
c2  = damping.strut_rear;

playback_speed = 0.1;
tF      = 1;
fR      = 30/playback_speed;
dt      = 1/fR;
time    = linspace(0,tF,tF*fR);

X_r = road.X_r;
Z_r = road.Z_r;

M = [   m   0   0   0   ;
        0   Iy  0   0   ;
        0   0   m1  0   ;
        0   0   0   m2  ];

C = [   c1+c2           a2*c2-a1*c1         -c1         -c2     ;
        a2*c2-a1*c2     c1*a1^2+c2*a2^2     a1*c1       -a2*c2  ;
        -c1             a1*c1               c1          0       ;
        -c2             -a2*c2              0           c2      ];

K = [   k1+k2           a2*k2-a1*k1         -k1         -k2     ;
        a2*k2-a1*k1     k1*a1^2+k2*a2^2     a1*k1       -a2*k2  ;
        -k1             a1*k1               k1+kt1      0       ;
        -k2             -a2*k2              0           k2+kt2  ];

F = [   0       0   ;
        0       0   ;
        kt1     0   ;
        0       kt2 ];

% State space model
A = [   zeros(4,4)      eye(4,4)    ;
        -M\K         -M\C    ];
B = [   zeros(4,2)  ;
        M\F      ];
C = [   1 0 0 0 0 0 0 0 ;
```

```
        0 1 0 0 0 0 0 0 ;
        0 0 1 0 0 0 0 0 ;
        0 0 0 1 0 0 0 0 ;
        0 0 0 0 0 0 0 0 ;
        0 0 0 0 0 0 0 0 ;
        0 0 0 0 0 0 0 0 ;
        0 0 0 0 0 0 0 0 ];
D = zeros(8,2);

sys = ss(A,B,C,D);

% Input
lon_pos_2 = vel*time + 0.5*acc*(time.*time);           % Longitudinal position of the↙
rear axle    [m]
lon_pos_1 = lon_pos_2 + a1+a2;   % Longitudinal position of the front axle   [m]
% OBS: Added wheelbase!
%
u1 = interp1(X_r,Z_r,lon_pos_1);
u2 = interp1(X_r,Z_r,lon_pos_2);

u_vet = [u1' u2'];

[y,time,x] = lsim(sys,u_vet,time);

z       = y(:,1); % Body vertical motion coordinate         [m]
theta   = y(:,2); % Body pitch motion coordinate            [rad]
zu1     = y(:,3); % Front wheel vertical motion coordinate  [m]
zu2     = y(:,4); % Rear wheel vertical motion coordinate   [m]

% Time step
dt = mean(diff(time));

% Velocity calculation
v_z = diff(z) / dt;  % Body vertical velocity
v_theta = diff(theta) / dt;  % Body pitch velocity
v_zu1 = diff(zu1) / dt;  % Front wheel velocity
v_zu2 = diff(zu2) / dt;  % Rear wheel velocity

% Acceleration calculation
a_z = diff(v_z) / dt;  % Body vertical acceleration
a_theta = diff(v_theta) / dt;  % Body pitch acceleration
a_zu1 = diff(v_zu1) / dt;  % Front wheel acceleration
a_zu2 = diff(v_zu2) / dt;  % Rear wheel acceleration

% Time vector for velocity and acceleration (one element less due to differentiation)

displacement.z_body =z;
displacement.z_unsprung_front = zu1;
displacement.z_unsprung_rear = zu2;
displacement.theta = theta;
displacement.time = time;
displacement.tire_front = u1/2;
displacement.tire_rear = u2/2;
displacement.longitudinal_pos_front = lon_pos_1;
displacement.longitudinal_pos_rear = lon_pos_2;

velocity.v_body = v_z;
velocity.v_unsprung_front = v_zu1;
velocity.v_unsprung_rear = v_zu2;
velocity.v_thetha = v_theta;
velocity.time = time(1:end-1);

acceleration.a_body = a_z;
acceleration.a_unsprung_front = a_zu1;
acceleration.a_unsprung_rear = a_zu2;
acceleration.a_theta = a_theta;
acceleration.time = time(1:end-2);

end
```