Midnight Sun Solar Car Team
University of Waterloo

FEA Boundary Conditions

MSXII

Prepared by:
Devon Copeland

September 26, 2017

1 Background

1.1 Vehicle Design

Midnight Sun Twelve (MSXII) is a cruiser class, solar electric vehicle being designed with the goal of competing in the 2018 American Solar Challenge (ASC 2018) and the 2019 World Solar Challenge (WSC 2019). By definition, cruiser class solar vehicles must be multi-occupant and are designed with the intent of being more practical than a typical, challenger class solar car. Because of the unique requirements of this class, the spring rates and target damping coefficients on MSXII's suspension must be selected to optimize for efficiency while still keeping driver comfort in mind. This report proposes an analytical technique for determining the response of a suspension system to any given road conditions.

1.2 Important Vehicle Parameters

Table 1 lists vehicle parameters that are of importance to the design and analysis of MSXII's suspension. The mass and moment of inertia was computed by CAD software.

Table 1: Important Vehicle Parameters

Mass (M)	550	kg
Wheelbase	2.60	m
Track	1.60	m
Distance from Front Wheel to CoG (a)	1.32	m
Distance from Road to CoG (h)	0.52	m
Moment of Inertia Normal to Symmetry Plane at CoG (I_{xx})	550	kgm^2
Full Wheel Travel in Compression	45	mm
Front Coilover Angle from Vertical	40	$\circ$
Rear Coilover Angle from Vertical	30	$\circ$

1.3 Suspension Architecture

MSXII's front suspension comprises of a double wishbone linkage with an outboard coilover while the rear suspension comprises of independent trailing arms allowing for zero scrub and thus less rolling resistance. Figure 1 shows the screenshots of the front and rear suspension.

(a) Front Isometric View (b) Rear Isometric View (c) Rear Sie View

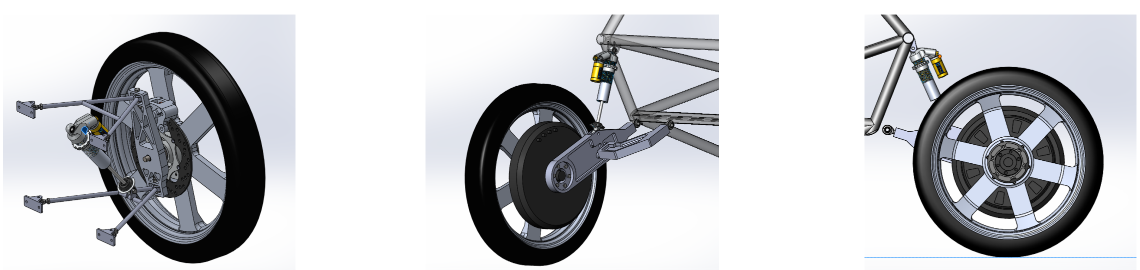

Figure 1: MSXII Suspension Design (Screenshots from Solidworks)

1.4 Two Loading Conditions

Two cases of boundary conditions are considered for FEA. One case occurs when the suspension is statically loaded during the extreme case of braking while cornering. The other case is a dynamic loading scenario where the car suddenly hits a bump at speed.

It is important to note that the former case (that of braking while cornering) is specific to the front suspension. The equivalent worst case loading scenario for the rear would occur while accelerating during a corner. However, since MSXII's motors produce only ***Nm of torque while the brakes require ***Nm to slow the car at the required rate, the worst loading for the front will far exceed that of the rear. For simplicity, this value is treated as the worst case for both the front and the rear.

2 Static Loading

Assuming a smooth driving surface, the largest normal force acting on the tire will occur while braking in a turn. To approximate this loading condition, the a superposition of two static half car models is used; one for cornering and one for hard braking.

2.0.1 Hard Braking

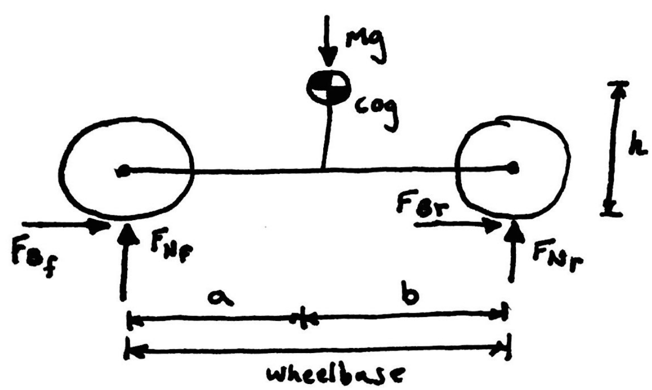

Figure 2: Free Body Diagram of a Car Braking

Figure 2 shows a free body diagram of a half car model undergoing braking. MSXII will only have brake callipers on the front wheels however regenerative braking from the hub motors will also provide a braking force at the rear contact patches. Assuming a generous coefficient of static friction, μ_s, of 1.0 and the extreme case where the both the font and rear tires are about to begin sliding, the combined braking force can be expressed as:

$$F_{braking} = F_{Nf} + F_{Nr} = \mu_s F_{N_{net}} = \mu_s M g = (1.0)(550kg)\left(9.81\frac{m}{s^2}\right) = 5.40kN \qquad (1)$$

From Figure 2, summing the moments about the centre of gravity and the vertical forces results in the following equations:

$$aF_{Nf} = hF_{braking} + bF_{Nr} \qquad (2)$$

$$F_{Nf} + F_{Nr} = Mg \tag{3}$$

Solving for the front normal force:

$$\begin{aligned} aF_{Nf} &= hF_{braking} + b(Mg - F_{Nf}) \\ F_{Nf} &= \frac{hF_{braking} + bMg}{a+b} = \frac{(0.52m)(5.40kN) + (1.28m)(550kg)\left(9.81\frac{m}{s^2}\right)}{(1.32m)+(1.28m)} = 3.74kN \end{aligned} \tag{4}$$

Note that 3.74kN is for both front wheels. Therefore there is approximately **1.87kN** of normal force on each front tire during hard braking.

2.0.2 Cornering

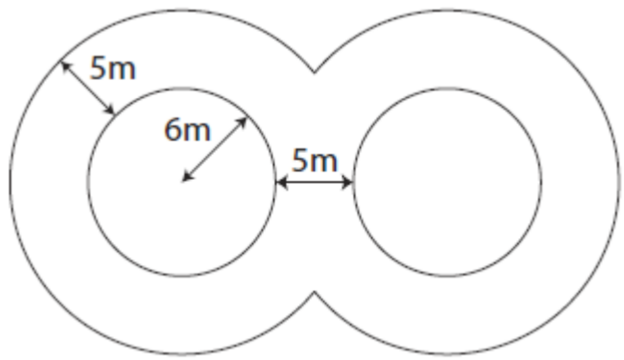

Figure 3: Figure of Eight Course from ASC 2018 Regulations [1]

The ASC 2018 regulations stipulate that a vehicle must be able to navigate a figure of eight as shown in Figure 3 in less than 18 seconds [1]. Since the average arc length of the figure of eight is $34\pi m$, the net cornering force required to navigate the figure of eight can be found as follows:

$$F_{si} + F_{so} = F_{steer} = \frac{Mv^2}{r} = \frac{(550kg)\left(\frac{34\pi m}{18s}\right)^2}{8.5m} = 2.28kN \tag{5}$$

For the half car model shown if Figure 4, summing the moments about the centre of gravity and the vertical forces results in the following equations:

$$\frac{track}{2}F_{No} = hF_{steer} + \frac{track}{2}F_{Ni} \tag{6}$$

$$F_{No} + F_{Ni} = Mg \tag{7}$$

Solving for the outside wheel normal force:

$$\begin{aligned} \frac{track}{2}F_{No} &= hF_{steer} + \frac{track}{2}(Mg - F_{Ni}) \\ F_{No} &= \frac{hF_{steer} + \frac{track}{2}Mg}{track} = \frac{(0.52m)(2.28kN) + (0.80m)(550kg)\left(9.81\frac{m}{s^2}\right)}{1.60m} = 3.44kN \end{aligned} \tag{8}$$

Note that the 3.44kN above is for both front and rear outside wheels. Assuming the force is evenly split between the front and rear, there is approximately **1.72kN** of normal force on each outside tire during maximum cornering.

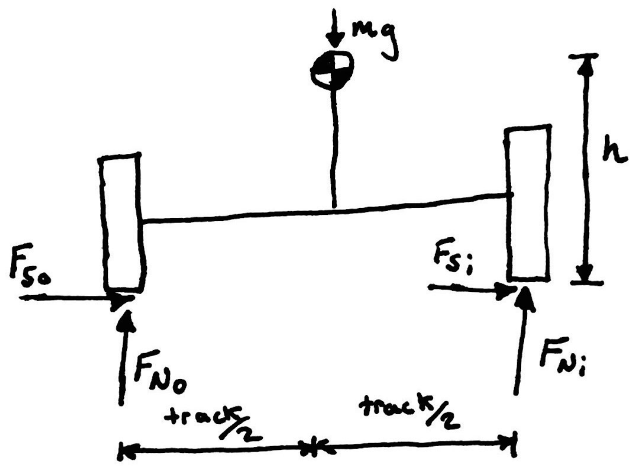

Figure 4: Free Body Diagram of a Car Cornering

2.0.3 Superposition

Both the braking and cornering conditions subject the front outside wheel to a normal force greater than the nominal normal force seen when the car is at rest. The change in normal force for each loading condition can be expressed as follows:

$$\begin{aligned}\Delta F_{N_{braking}} &= F_{Nf_{braking}} - \frac{Mg}{4} = 1.87kN - 1.35kN = 0.52kN \\ \Delta F_{N_{cornering}} &= F_{No_{cornering}} - \frac{Mg}{4} = 1.72kN - 1.35kN = 0.37kN\end{aligned} \tag{9}$$

To approximate the worst case loading condition, a superposition of the changes in normal force and the nominal normal force is applied.

$$F_{N_{max}} = \Delta F_{N_{braking}} + \Delta F_{N_{cornering}} + F_{N_{nominal}} = 0.52kN + 0.37kN + 1.35kN = 2.24kN \tag{10}$$

3 Dynamic Loading

MSXII will be using Ohlins TTX25 dampers for both the front and rear suspension. From the Ohlins' published data, [?], the maximum damping coefficient achievable is approximately 3.5kNs/m.

References

[1] American Solar Challenge. 2018 regulations, [online], 2017. http://americansolarchallenge.org/ASC/wp-content/uploads/2016/08/ASC2018-Regs-External-Revision-A.pdf.

A MATLAB Source Code for Simplified Model

```
% ----------------------------------------------------------------------- %
%
% Suspension Spring Rates Project - alternativeFormulation.m
% Function to setup transfer functions for a half car model to determine the
% reponse of MSXII to disturbances in the road conditions
%
% Author: Devon Copeland
%
%   Midnight Sun 2017
%
%
% Notes:
%
%   -
% ----------------------------------------------------------------------- %

load_system('halfCarModel');
close all;
clear all

% ------------------------------ Constants ------------------------------- %

c1_ = 4830;      % Front damping coefficient (Ns/m)
c2_ = 4830;      % Rear damping coefficient (Ns/m)
k1_ = 600*175.1*cosd(35);    % Front sring rate (N/m)
k2_ = 500*175.1*cosd(30);    % Rear sring rate (N/m)
m_ = 550;        % Mass of the vehicle (kg)
I_ = 550;        % Moment of interial about global x axis (kg*m^2)
a_ = 1.32;        % Distance from COG to front tire (m)
b_ = 1.28;        % Distance from COG to rear tire (m)

% ---------------------- Forcing Function Parameters -------------------- %

amplitude = 0.015;               % Amplitude of bumps on road surface
bumpSparation = 2;               % Distance between peaks on the road surface (m)
speed = 22.2;                    % Cruising speed (m/s)
wb = 2*pi*speed/bumpSparation;   % Frequency of base excitation (rad/sec)
timeDelay = (a_+b_)/speed;       % delay between input striking front and
                                 % rear wheels (s)

% Update simulink parameters
set_param('halfCarModel/DelayA','DelayTime',mat2str(timeDelay));
set_param('halfCarModel/DelayB','DelayTime',mat2str(timeDelay));

% ----------------- Create Symbolic Transfer Functions ----------------- %

syms t s c1 c2 k1 k2 m I a b t0 A w0 c k

QA1 = m*s^2 + (c1+c2)*s + (k1+k2); % X terms of ODE (A)
QA2 = (b*c2-a*c1)*s + (b*k2-a*k1); % Theta terms of ODE (A)
QB1 = (b*c2-a*c1)*s + (b*k2-a*k1); % X terms of ODE (B)
QB2 = I*s^2 + (b^2*c2+a^2*c1)*s + (b^2*k2+a^2*k1); % Theta terms of ODE (B)
```

```
PA1 = (c1*s+k1); % Non time shifted componet of P for ODE (A) (front tire)
PA2 = (c2*s+k2); % Time shifted componet of P for ODE (A) (rear tire)
PB1 = (-a*c1*s-a*k1); % Non time shifted componet of P for ODE (B) (front tire)
PB2 = (b*c2*s+b*k2); % % Time shifted componet of P for ODE (B) (rear tire)

% Create linear systems in to solve for transfer functions

Q = [QA1, QA2;
     QB1, QB2];

P1 = [PA1;
      PB1];
P2 = [PA2;
      PB2];

H1 = linsolve(Q,P1);
H2 = linsolve(Q,P2);

% Input in frequency domain
phaseShift = exp(-s*timeDelay); % Time shift to be applied to P_2
input = (amplitude*wb)/(s^2+wb^2); % Laplace transform of input L{A*sin(wb*t))}

% -------------------------- Solve for response  ----------------------- %

simplifiedH1 = simplify(collect(subs(H1, {k1 k2 c1 c2 m I a b}, {k1_ k2_ c1_ c2_ m_ I_ a_ b_})));
simplifiedH2 = simplify(collect(subs(H2, {k1 k2 c1 c2 m I a b}, {k1_ k2_ c1_ c2_ m_ I_ a_ b_})));
impulseResponse = vpa(simplify(ilaplace(simplifiedH1) + ilaplace(simplifiedH2*phaseShift)),3);

Fs = 200; % sampling frequency in Hz
T = 1/Fs; % sampling period
t_ = 0:T:30; % time values 0 to 30 seconds
L = length(t_);

% Numerical impulse response with zero damping
numericalImpulseResponse = double(subs(impulseResponse,t,t_));
figure;
hold on;
plot(t_,numericalImpulseResponse(1,:).*1000,'DisplayName','x_0');
ylabel('Linear Displacement (mm)');
yyaxis right
plot(t_,numericalImpulseResponse(2,:).*(180/pi),'DisplayName','\theta');
ylabel('Angular Displacement (deg)');
xlabel('Time (s)');
xlim([0 1]);
title('Impusle Response');
legend show;
saveas(gcf,'impResp','png');

% Compute the first harmonic of the displacement
fftOut = fft(numericalImpulseResponse(1,:));
p2 = abs(fftOut/L);
p1 = p2(1:L/2+1);
```

```
p1(2:end-1) = 2*p1(2:end-1);
f = Fs*(0:(L/2))/L;
figure;
hold on
plot(f,p1,'DisplayName','x_0');

% FourierTransform
fftOut = fft(numericalImpulseResponse(2,:));
p2 = abs(fftOut/L);
p1 = p2(1:L/2+1);
p1(2:end-1) = 2*p1(2:end-1);
f = Fs*(0:(L/2))/L;
plot(f,p1,'DisplayName','\theta');

legend show
xlabel('Frequency (Hz)');
title('FFT of Impusle Response');
xlim([0 10]);
saveas(gcf,'fftImpResp','png');

% --------------------------------- Forces ----------------------------- %


% ------------------------------- Simulink ----------------------------- %

numericalH1 = subs(H1, {c1 c2 k1 k2 m I a b}, {c1_ c2_ k1_ k2_ m_ I_ a_ b_});
numericalH2 = subs(H2, {c1 c2 k1 k2 m I a b}, {c1_ c2_ k1_ k2_ m_ I_ a_ b_});

[numH1, denomH1] = numden(numericalH1);
[numH2, denomH2] = numden(numericalH2);

transFuncA1 = tf(sym2poly(numH1(1)),sym2poly(denomH1(1)));
set_param('halfCarModel/HA1','Numerator',mat2str(sym2poly(numH1(1)))...
                            ,'Denominator',mat2str(sym2poly(denomH1(1))));

transFuncA2 = tf(sym2poly(numH2(1)),sym2poly(denomH2(1)));
set_param('halfCarModel/HA2','Numerator',mat2str(sym2poly(numH2(1)))...
                            ,'Denominator',mat2str(sym2poly(denomH2(1))));

transFuncB1 = tf(sym2poly(numH1(2)),sym2poly(denomH1(2)));
set_param('halfCarModel/HB1','Numerator',mat2str(sym2poly(numH1(2)))...
                            ,'Denominator',mat2str(sym2poly(denomH1(2))));

transFuncB2 = tf(sym2poly(numH2(2)),sym2poly(denomH2(2)));
set_param('halfCarModel/HB2','Numerator',mat2str(sym2poly(numH2(2)))...
                            ,'Denominator',mat2str(sym2poly(denomH2(2))));

% --------------- Solving for natural resonance frequency -------------- %
```

B MATLAB Source Code for Alternative Formulation

```
% ----------------------------------------------------------------------- %
%
% Suspension Spring Rates Project - simplifiedFormulation.m
% Function to setup simplified transfer functions for a half car model to
% determine the reponse of MSXII to disturbances in the road conditions
%
% Author: Devon Copeland
%
%   Midnight Sun 2017
%
%
% Notes:
%
%   -
% ----------------------------------------------------------------------- %

close all;
clear all;
clc;

% ----------------------------- Constants ------------------------------ %

c_ = 4830;      % Front damping coefficient (Ns/m)
k_ = 88600;     % Front sring rate (N/m)
M_ = 550;       % Mass of the vehicle (kg)
I_ = 550;       % Moment of interial about global x axis (kg*m^2)
a_ = 1.3;       % Distance from COG to front tire and rear tire (m)
zx_ = 0.49;      % Damping ratio of translational
wnx_ = 17.95;    % Natural frequency of pitch
zth_ = 0.64;     % Damping ratio of pitch
wnth_ = 23.33;   % Natural frequency of pitch


% ---------------------- Forcing Function Parameters ------------------- %

A_ = 0.015;                        % Amplitude of bumps on road surface
bumpSparation = 2;                 % Distance between peaks on the road surface (m)
speed = 22.2;                      % Cruising speed (m/s)
wb_ = 2*pi*speed/bumpSparation;    % Frequency of base excitation (rad/sec)
t0_ = (2*a_)/speed;               % delay between input striking front and
                                   % rear wheels (s)

% ----------------- Create Symbolic Transfer Functions ----------------- %

syms t s M I a t0 A wb c k zx wnx zth wnth

QA = s^2 + 2*c/M*s + wnx^2; % ODE A
QB = s^2 + 2*a^2*c/I*s + wnx^2; % ODE B

phaseShift = exp(-s*t0);

PA = [(c*A*wb/M)*(s/(s^2+wb^2));
```

```
      (c*A*wb/M)*(s/(s^2+wb^2))*phaseShift;
      (k*A/M)*(wb/(s^2+wb^2));
      (k*A/M)*(wb/(s^2+wb^2))*phaseShift];

PB = [(-c*a*A*wb/I)*(s/(s^2+wb^2));
      (c*a*A*wb/I)*(s/(s^2+wb^2))*phaseShift;
      (-k*a*A/I)*(wb/(s^2+wb^2));
      (k*a*A/I)*(wb/(s^2+wb^2))*phaseShift];

X = collect((PA./QA),s);
Theta = collect((PB./QB),s);

subX = simplify(subs(X, {M I a t0 A wb c k zx wnx zth wnth},{M_ I_ a_ t0_ A_ wb_
    c_ k_ zx_ wnx_ zth_ wnth_}));
subTheta = simplify(subs(Theta, {M I a t0 A wb c k zx wnx zth wnth},{M_ I_ a_ t0_
     A_ wb_ c_ k_ zx_ wnx_ zth_ wnth_}));

x = simplify(collect(sum(ilaplace(subX))));
theta = simplify(collect(sum(ilaplace(subTheta))));

% Print total response
latex(vpa(x,3))
latex(vpa(theta,3))

totalResponse = [x;theta];

% Plot total response
t_ = 0:0.005:1;
numericalResponse = double(subs(totalResponse,t,t_));
figure;
hold on;
plot(t_,numericalResponse(1,:).*1000,'DisplayName','x_0');
ylabel('Linear Displacement (mm)');
yyaxis right
plot(t_,numericalResponse(2,:).*(180/pi),'DisplayName','\theta');
ylabel('Angular Displacement (deg)');
xlabel('Time (s)');
xlim([0 1]);
title('Total Response');
legend show;
saveas(gcf,'totalResp','png');
```