Designing a Lap Simulator for the Shell Eco-Marathon Intermediate Report

Supervisor: Tim Baker
Student: Hasha Dar

University College London — January 21, 2022

1 Project Background & Problem

The Shell Eco-Marathon is a competition at the school and university level for students focusing on energy optimisation in vehicles. The aim of the competition is to develop new innovations in energy efficiency for vehicles on the road with the idea of reducing carbon emissions (Shell Eco-Marathon 2021). I will be focusing on the 'Prototype' category for the competition, which involves designing a hydrogen-powered single-seater vehicle with a focus on attaining ultra-efficiency through the optimisation of the aerodynamic and performance characteristics of the vehicle.

For an ultra-energy-efficient vehicle, there are many sources of inefficiency. Some of them are aerodynamic drag (air resistance), friction drag (from the road surface) and losses in the powertrain (electricity generation, mechanical losses) (Wei et al. 2019). The design itself of the car must be optimised to increase its performance such as the weight of the vehicle (Tsirogiannis et al. 2019).

A problem that arises when investigating and modelling these variables is that there is no way of testing certain performance characteristics within the holistic context of the competition virtually. Currently, to test newly designed components and its overall impact on the performance of the vehicle, the part must be prototyped and installed on the vehicle, and then experimentally tested. This presents a problem as this is costly, time-consuming and constrains the number of prototypes that can be made.

A transient, dynamic lap simulator will allow the analysis of the vehicle's various performance characteristics. This will allow the team to optimise and prototype designs much more effectively. The team would be able to test the vehicle on the final test track, without the cost of being there physically.

2 Project Aim

During the design phase of the vehicle, a lot of work must go into modelling performance variables of the vehicle. This may be done through mathematical analysis and through the use of software such as Computational Fluid Dynamics (CFD) and Finite Element Analysis (FEA). These methods allow us to test whether our design meets our performance criteria (in a virtual sense). This project will focus on bringing together the various performance characteristics of the vehicle to be able to analyse the car in a contextual manner. A lap simulator will allow the team to see how their prototype design will function in a real-world scenario, rather than as a discrete set of performance variables. Hence, with a lap simulator the team will be able to:

- unify the various computational models that assess parts of the vehicles performance
- see where the primary energy inefficiencies are in a transient, dynamics test, allowing targeted improvement of the vehicle's performance
- reduce the number of physical tests for the vehicle,
  - lowering cost
  - reducing development time
  - minimising the number of prototype parts manufactured

3 Methodology

1. Research and select a race track for modelling
2. Build racetrack analysis code (to find turning angles at any moment)
3. Research vehicle performance parameters (motor, drivetrain, vehicle)
4. Integrate vehicle performance parameters into MATLAB model
5. Integrate vehicle constraints (friction limits, cornering force, electrical power)
6. Design Simulink model to link performance parameters together
7. Design motor control loops (dependent on race strategy)
8. Generate vehicle data (displacement, velocity, acceleration)
9. Generate lap time for vehicle
10. Optimise race strategy to improve lap time
11. Improve fidelity of Simulink model
12. Develop GUI for simulation

3.1 Track data and vehicle parameters

The following MATLAB code takes racetrack racing line coordinate data and turns adjacent points into vectors. The angle between adjacent vectors are then calculated and stored in an array. This gives us the instantaneous racing line track curvature at any point. Next, we see that the performance parameters for the various vehicle subsystems are stored.

```
clc
clear
close all

%pull racing line data
raceTrackXY = readmatrix('brandsHatchRaceLineXYData.csv');
%plot(raceTrackXY(:,1), raceTrackXY(:,2))

%calculate turning angle
%initialise arrays
u = zeros(numel(raceTrackXY(:,1))-1,numel(raceTrackXY(1,:)));
v = u;
a = zeros(numel(raceTrackXY(:,1))-1,1);
%generate vectors between 2 adjacent track points
for i = 1:(numel(raceTrackXY(:,1))-1)
    u(i,:) = [(raceTrackXY(i+1,1) - raceTrackXY(i,1)), (raceTrackXY(i
        +1,2)-raceTrackXY(i,2))];
end
temp = numel(raceTrackXY(:,1));
%generate vector one step ahead of u
for i = 2:(temp - 1)
        v(i,:) = u((i-1),:);
end
%clean up vectors and add temp z data
v(1,:) = u(temp-1,:);
u = cat(2,u,a);
v = cat(2,v,a);
%initialise array for angle
angleTheta = zeros((temp - 1),1);
```

```
%generate angles
for i = 1:temp-1
    angleTheta(i) = subspace(u(i,:)', v(i,:)');
    angleTheta(i) = angleTheta(i)*57.2958; %convert to degrees
end

%plot(linspace(0,4206,numel(angleTheta)),angleTheta)

%define time
t = linspace(0,10,0.01)';

%motor parameters
motor.NMaxTq = 0; %motor speed for max torque (rpm)
motor.NMaxPwr = 500; %motor speed for max power (rpm)
motor.MaxPower = 1000; %motor max rated power (W)
motor.MaxTorque = 10; %motor rated torque (Nm)
motor.NtqFullLoad = linspace(0,1000,100); %motor speed axis (rpm)
motor.tqFullLoad = [linspace(0,100,100); linspace(10,0,100)]'; %motor
    torque curve at full load (Nm)
motor.Nmax = 1000; %max motor speed (rpm)
motor.Nmin = 0; %min motor speed (rpm)
for i = 1:numel(motor.NtqFullLoad)
    motor.pwrFullLoad(i) = -(motor.MaxTorque/motor.Nmax)*(motor.
       NtqFullLoad(i))^2 + motor.MaxTorque*motor.NtqFullLoad(i);
end

%transmission
gearbox.i0 = 5; %final drive ratio at the differential
gearbox.effic = 0.9; %drivetrain efficiency

%tyre spec
tyre.width = 0.05; %tyre width (m)
tyre.dia = 0.5; %tyre diameter (m)
tyre.miua = 0.7; %tyre friction coefficient
tyre.load = 0.7; %rear axle load coeffcient
tyre.rwd = 0.98*tyre.dia; %tyre diameter under load

%vehicle
veh.curbMass = 100; %curb mass (kg)
veh.driverMass = 60; %driver mass (kg)
veh.massFactor = 1.05; %mass factor
veh.g = 9.81; %grav constant (m/s^2)
veh.cd = 0.1; %drag coefficient
veh.fa = 0.5; %frontal area (m^2)
veh.rho = 1.225; %air density (kg/m^3)
road.cr = 0.011; %road load coefficient
veh.mass = veh.curbMass*veh.massFactor + veh.driverMass;
road.slope = 0;

input = 1000;
```

Next, a Simulink model was created to model the various performance parameters:

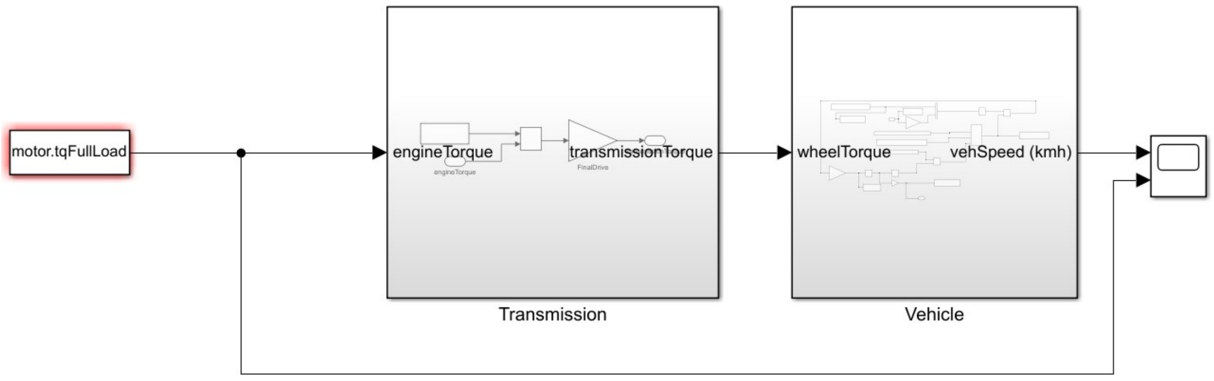

Figure 1: Current Simulink Model.

Here we can see that our motor torque (following a traditional DC motor power curve) is inputted into the transmission subsystem:

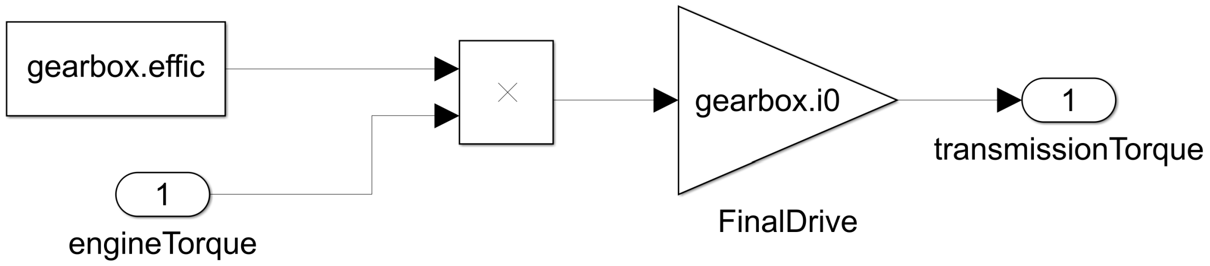

Figure 2: Drivetrain model.

Here, we see that the drivetrain ratios and efficiency is inputted. This outputs our transmission torque.

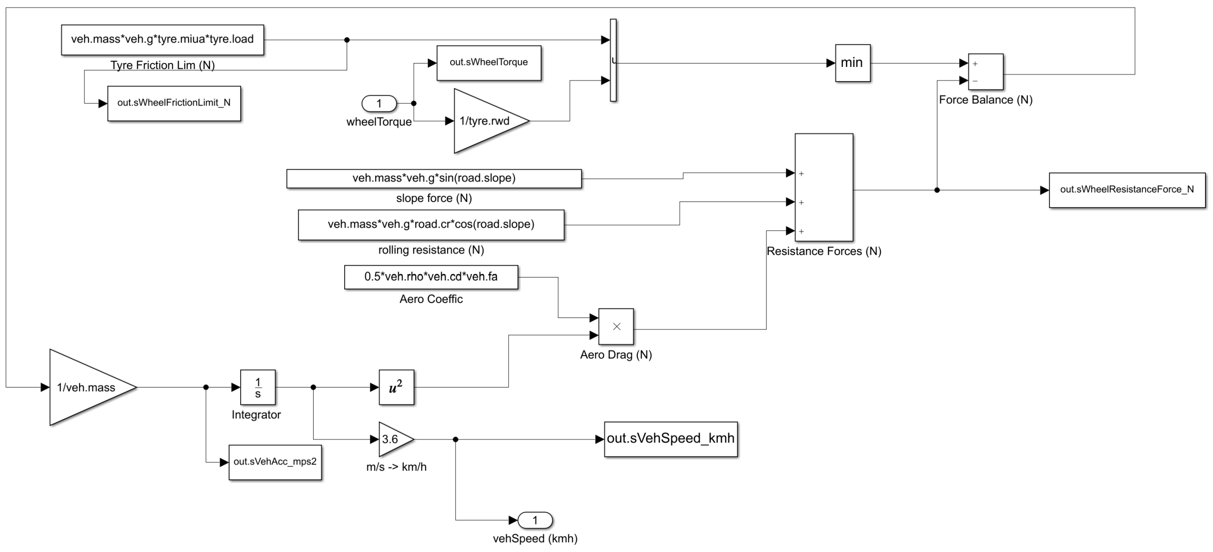

Figure 3: Vehicle model.

This subsystem starts by converting the transmission torque into the force exerted by the tyre onto the road. This is then cross-checked with the tyre friction limit with a ‘min’ statement (as the tyre friction limit cannot be exceeded.) The resistive forces are then summed (cornering force is yet to be modelled) and subtracted from the driving force. We can then calculate vehicle acceleration and velocity (which is looped back into our aerodynamic drag force).

The track data is also yet to be implemented and will be part of another subsystem, that will take in information such as displacement and output trajectory. The trajectory data will be used within this subsystem to calculate the limits on cornering force and whether the tyres start to slip.

Currently the motor torque has been modelled as a variable controlled by the user. The motor model being developed will fully simulate a 3-phase BLDC motor using PWM control. Thus, our only input to the system will become the throttle position. A model of the DC motor subsystem is shown below.

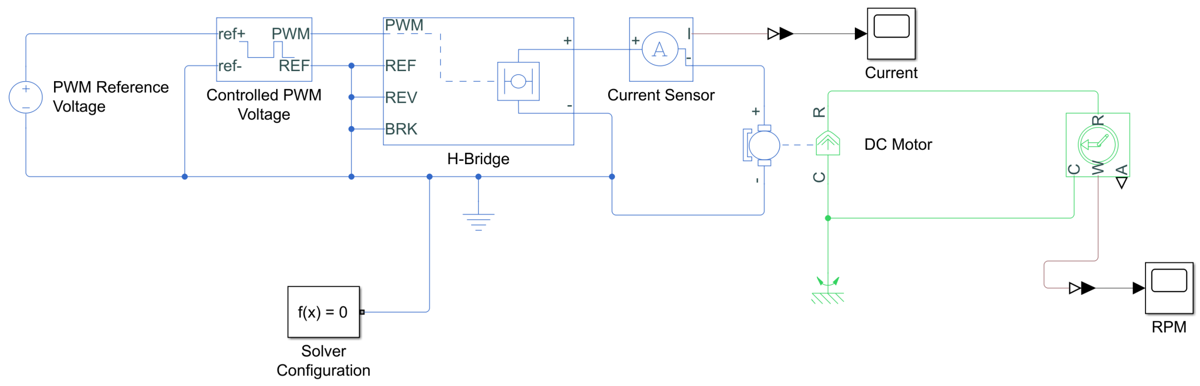

Figure 4: DC motor model.

4 Results

As a test, I inputted a simple step command as the motor torque (50 seconds on, 50 seconds off at $5\,\mathrm{N\,m}$). This yields the following straight line performance of the vehicle.

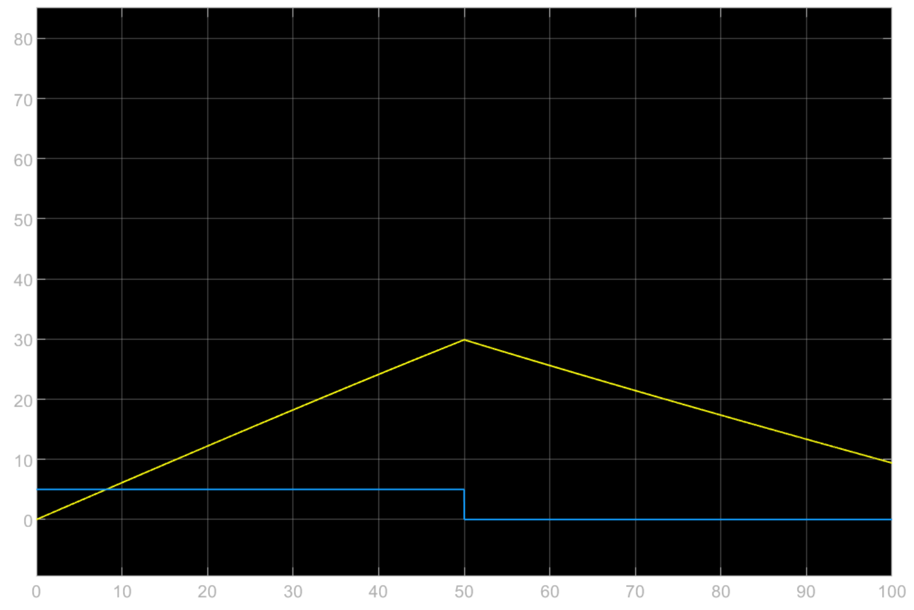

Figure 5: Straight Line Performance (x axis - time (s), y axis -velocity (m/s) and torque (Nm)).

Currently, the data has not been validated with empirical testing, but this is something I hope to do soon. The true values for the vehicle performance parameters are also to be confirmed, but have been ball-parked using online data and reports from previous years. Once, these are confirmed, the model can be evaluated against empirical data from the current Shell Eco Car.

5 Declaration

I, Hasha Dar, confirm that the work presented in this report is my own. Where information has been derived from other sources, I confirm that this has been indicated in the report.

References

Shell Eco-Marathon (2021), ‘Shell eco-marathon competiton - about’.
URL: *https://www.makethefuture.shell/en-gb/shell-eco-marathon*

Tsirogiannis, E. C., Stavroulakis, G. E. & Makridis, S. S. (2019), ‘Electric car chassis for shell eco marathon competition: Design, modelling and finite element analysis’, *World Electric Vehicle Journal* **10**.

Wei, Z., Song, Y., Wang, L., Zhang, C., Lam, J. C. C., Teng, X., Zhou, E. & Han, M. (2019), ‘Design and energy efficiency analysis of a pure fuel cell vehicle for shell eco racer’, *International Journal of Energy Research* **43**.