OptimumLap for dummies (and EEs)

version 0.1

Michal Podhradsky, michal.podhradsky@pdx.edu

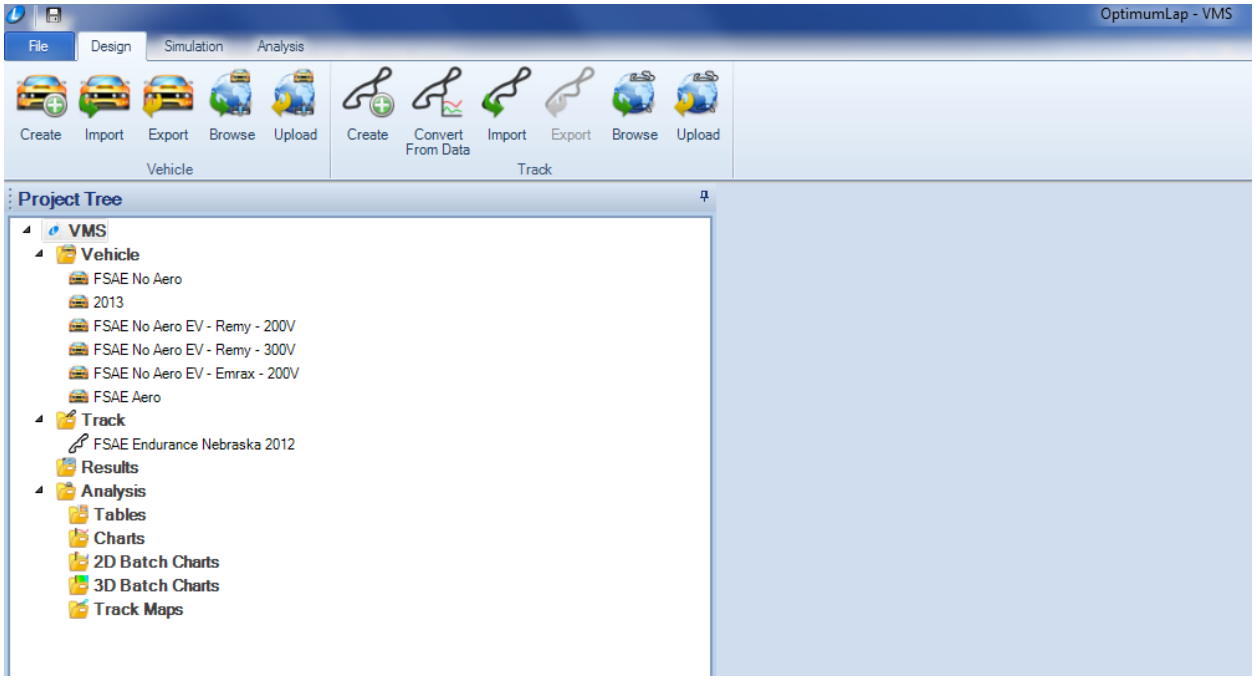
Introduction

This document covers how to use OptimumLap[1] for design of Electric Vehicle for FSAE competition. We cover as an example current configuration of our EV, and discuss the results. This is intended as a manual for anyone who wants to start with OptimumLap for EV.

Installation

OptimumLap can be downloaded for free from http://www.optimumg.com/software/optimumlap/ and it requires a Windows machine.[2]

Then unzip the project file (the project is called VMS), start OptimumLap and open the project.[3] You will see something like this:



[1] http://www.optimumg.com/software/optimumlap/

[2] or virtual-box with Windows, although that wasn't tested

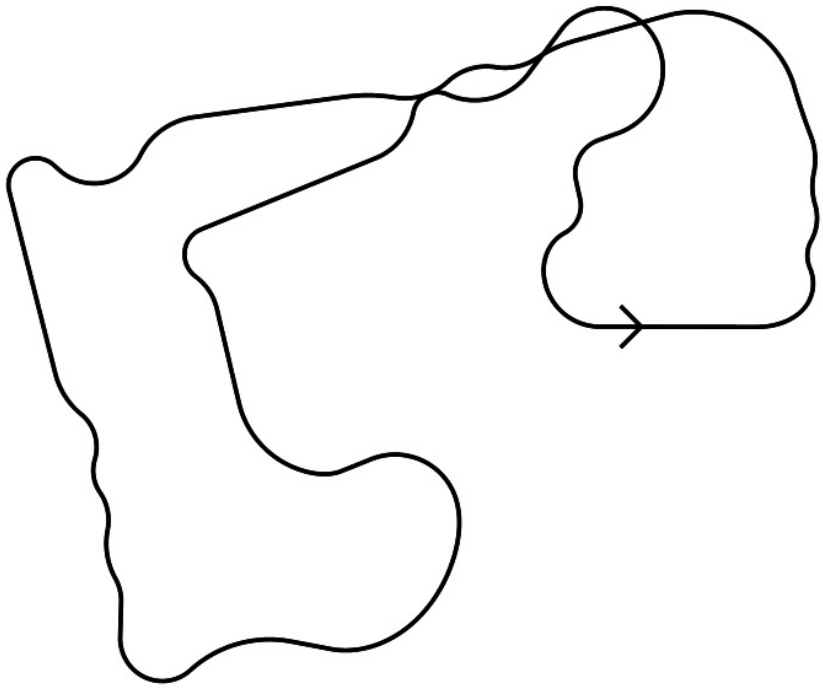
[3] in the future I am hoping to use some version-control-system so it is easier to be up to date with the project

First, the available vehicles are:

- FSAE No Aero - a sample vehicle that represents FSAE race car without Aero, can be downloaded from OptimumLap web, for reference only
- 2013 - the most recent vehicle I found on VMS drive, for reference only
- FSAE Aero - a sample vehicle with aero, for reference only
- **FSAE No Aero EV - Remy - 200V** - our current car configuration, we will be using this one
- other cars can be ignored

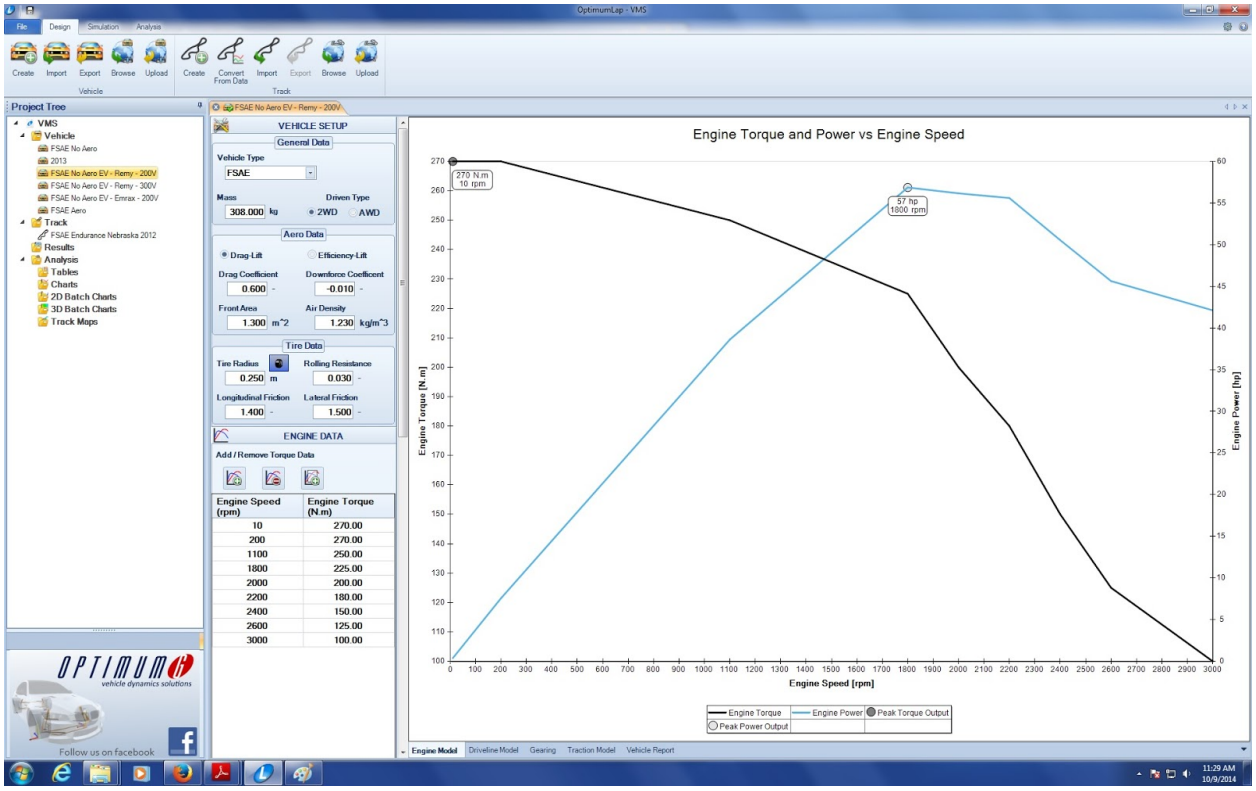
The track is set to FSAE Endurance Nebraska from 2012, because Endurance is most challenging for EVs and the tracks don't change too much from year to year.

FSAE Endurance Nebraska 2012, Lincoln - NE, USA



Vehicle

Use **FSAE No Aero EV - Remy - 200V,** click on the vehicle and you ll see this:



The configuration is:

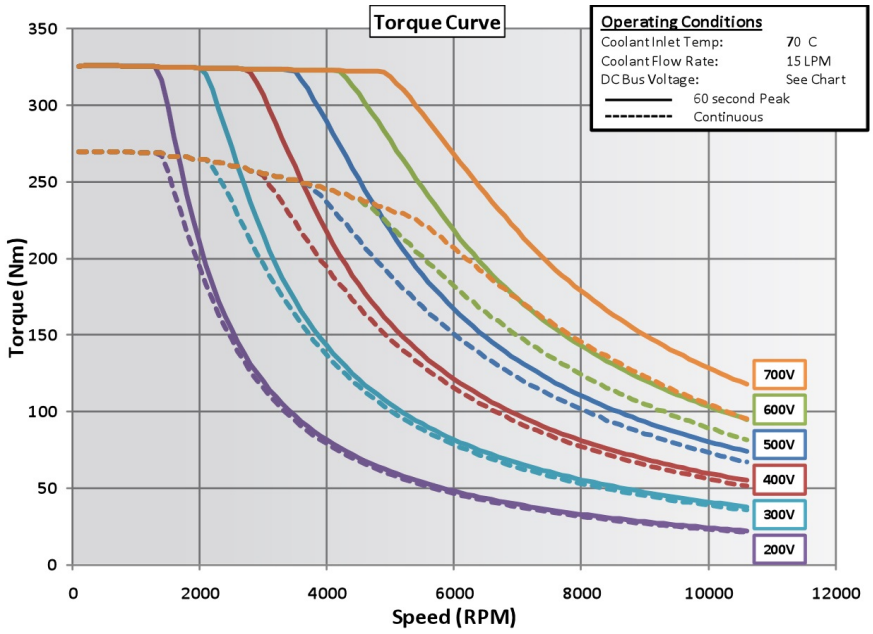
- 4x Enerdel Battery module (http://www.enerdel.com/mp320-049-moxie-battery-module/)
- Remy HVH250 motor
- Rinehart PX100 motor controller

Now the parameters that can be changed:

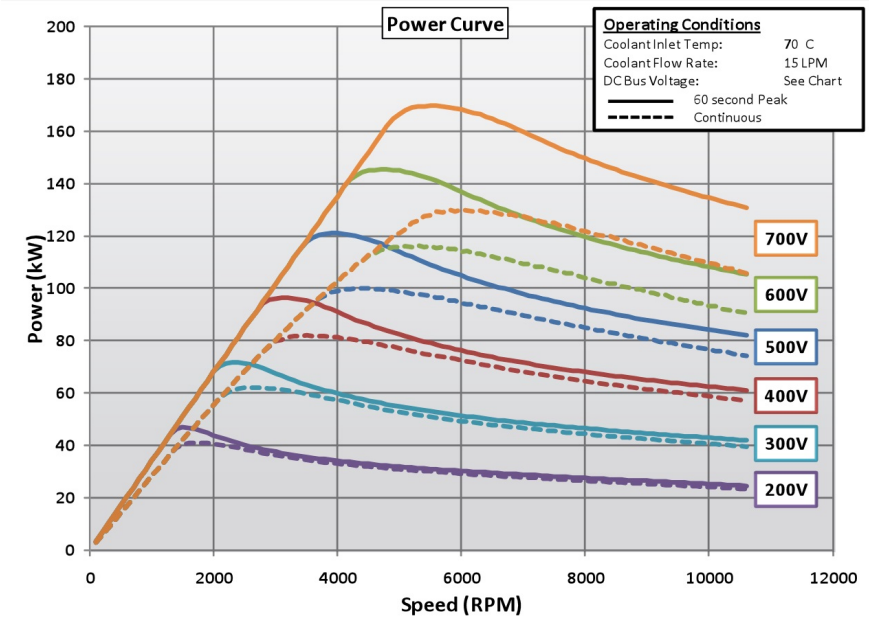
- weight - measured 308kg without driver
- Aero data - I left it the same as for the example car *FSAE No Aero*, because we don't know these at the moment
- Tire data - again, identical to *FSAE No Aero*
- Engine data - here comes the important part. i took the data from Remy datasheet (http://www.remyinc.com/docs/hybrid/REM-03_HVH250_DataSht_master.pdf) and filled in the values for torque manually.
- Thermal efficiency - 97% is a good value for an electric motor
- Fuel Energy density - changed to LithiumIon battery
- Transmission data - only one gear with gear ratio on 1 (direct drive)
- Final drive ratio - set to 3, because thats what I believe we currently have on the car
- Drive efficiency - again, 95% seems to be reasonable value
- Scaling factors - left to 100%

Engine data

We are operating the motor at 200V[4] so we use the 200V torque-RPM data from the motor datasheet. We are using continuous, not peak power at this point (somewhat conservative estimate).

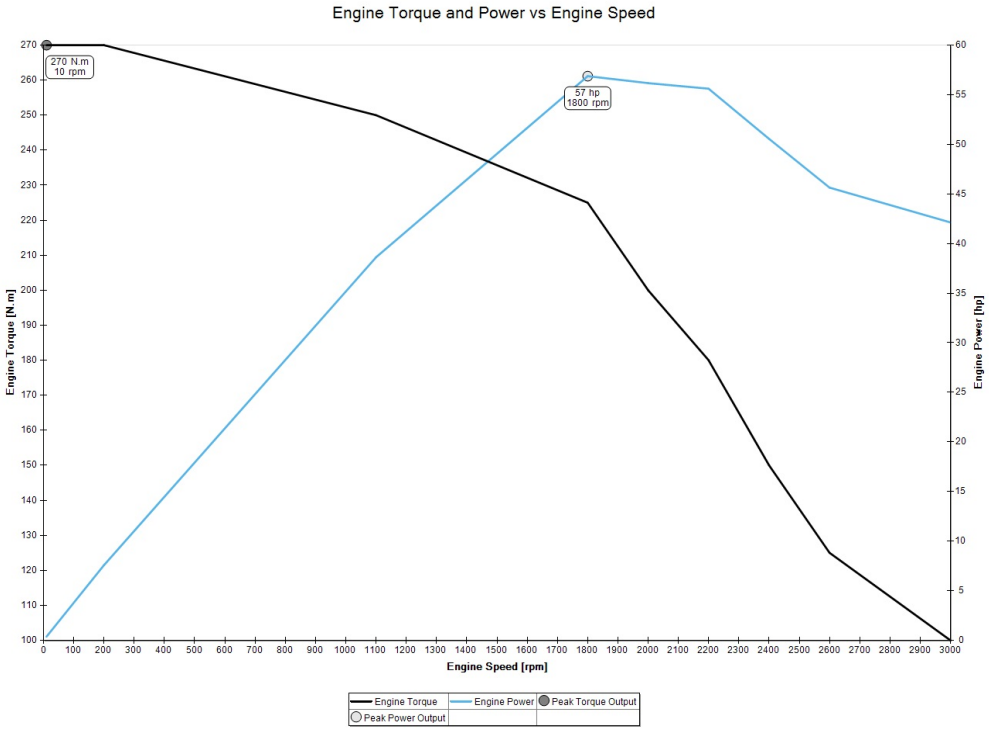


Although the motor data go up to 12000 RPM we cannot spin that fast because of the field weakening and the final drive ratio.[5] It turns out that it is not a big problem, because the peak power motor delivers is at around 1800 RPM.

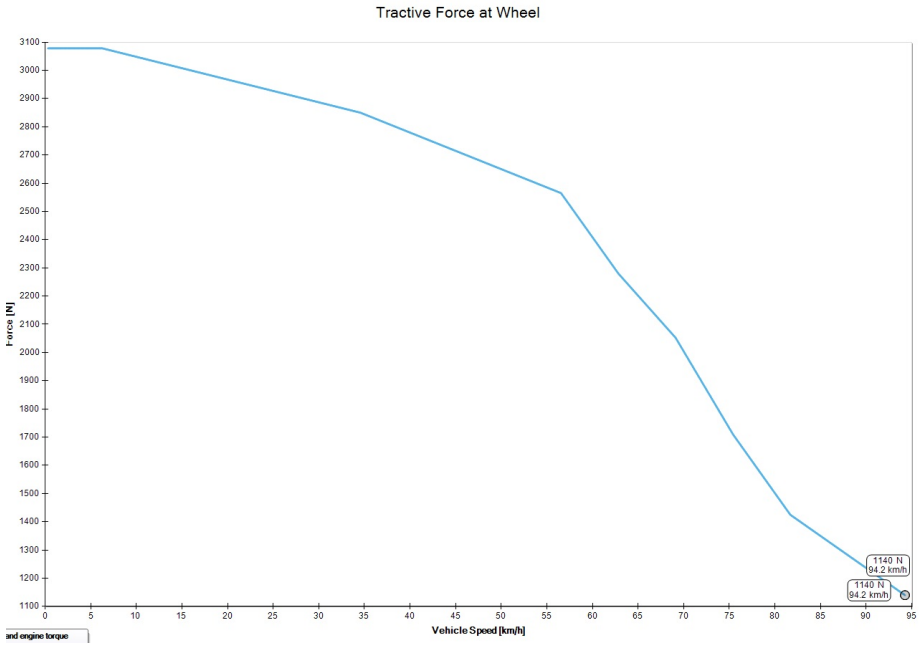


[4] it is actually 170V nominal, but that is not that important at this point
[5] we can't spin faster than 2500RPM

The datapoints are rather sparse, but we can approximate and type it in. The results match the datasheet, with peak torque at low-end of RPM and peak power at 1800 RPM.

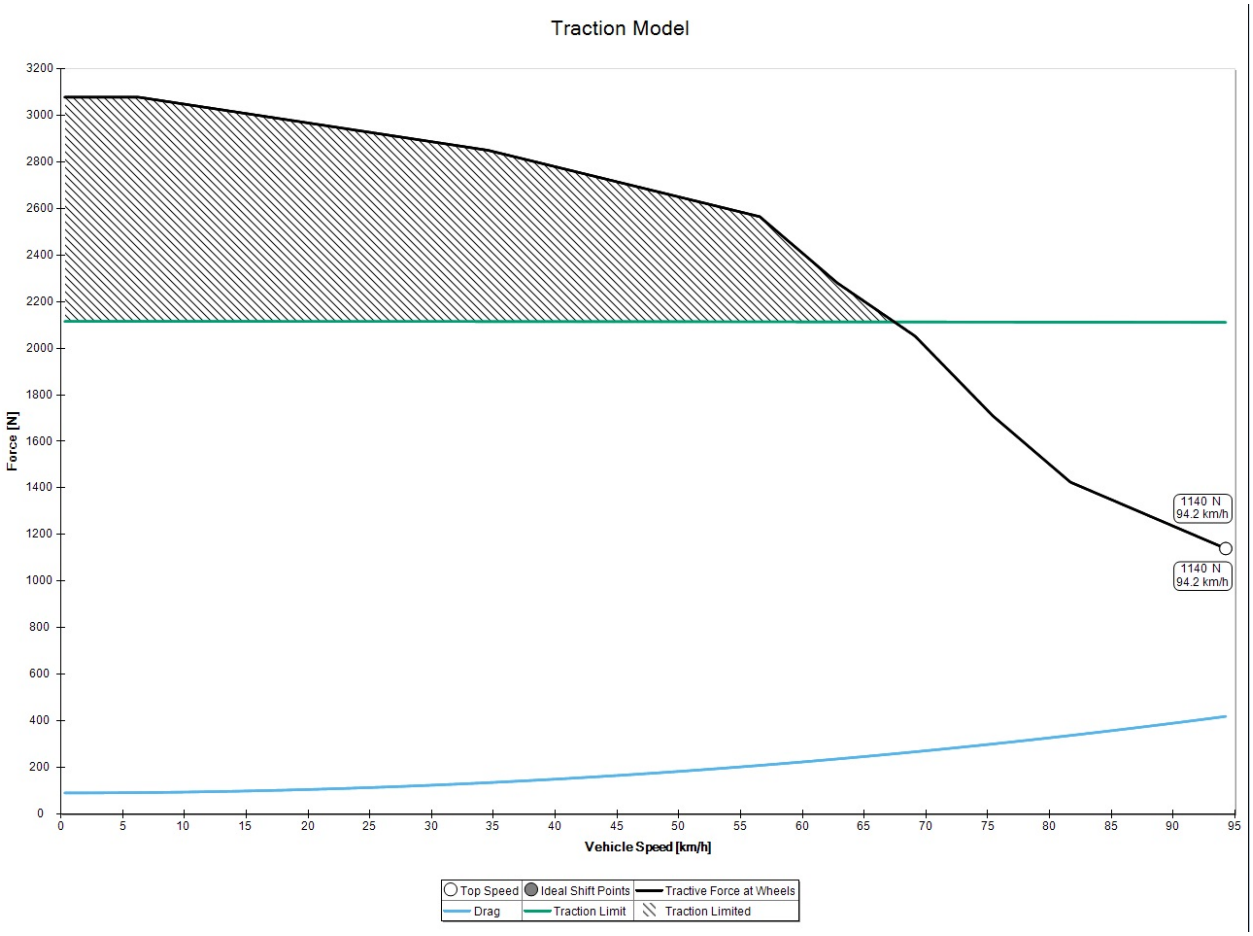


A sanity check - lets look at the driveline model, and see if it makes sense.



We can see that the max vehicle speed is 95 km/h (~60mph) which is what we calculated before with our current final drive ratio.

To be sure, lets have a look at the traction model. The faster we go, the higher is drag, which makes sense. The tractive force decreases with higher speed, which also generally makes sense (https://en.wikipedia.org/wiki/Tractive_force).

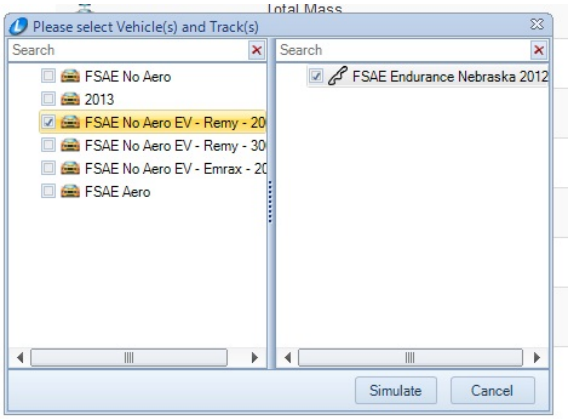


Variations

For the EV simulations, engine data is the part that is the most critical. Depending on our battery configurations, we will have different power curves for the motor, and that will affect the performance. The next big parameter is vehicle mass, dependent on batteries.

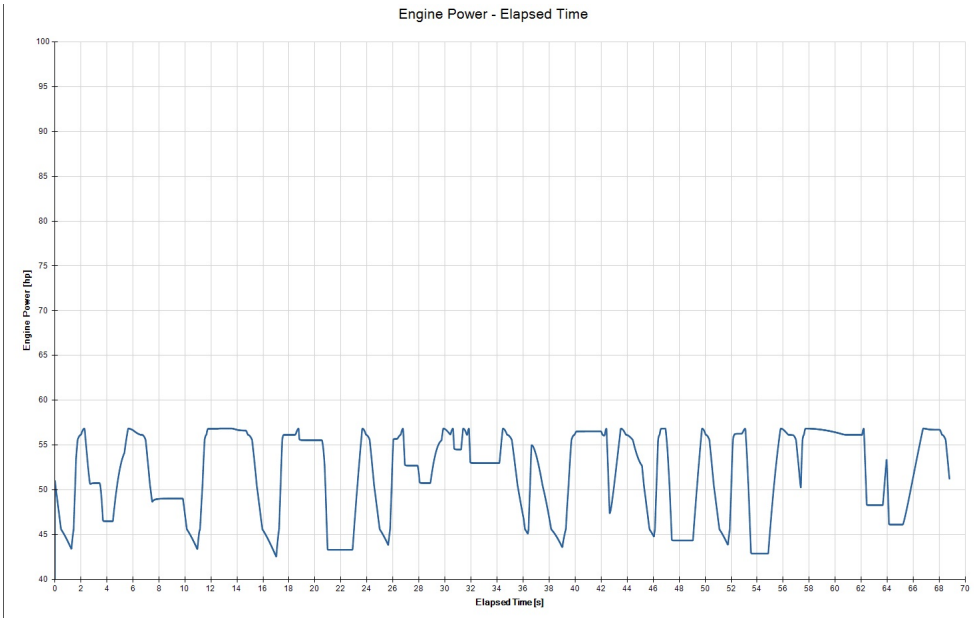
Simulation

Click on Simulation tab, and select **FSAE No Aero EV - Remy - 200V** and Simulate.



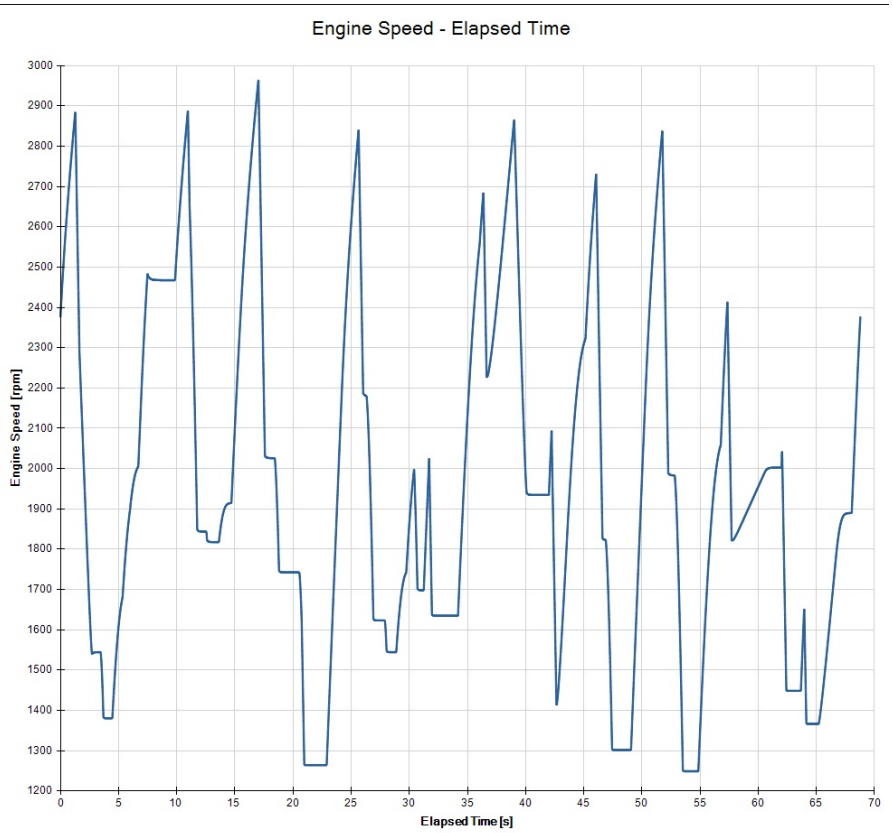
Now you ll get lap simulation results in the Results box and can use features from the Analysis tab. First important value is lap time. We got 68.78 seconds, which is reasonable, because last year the EVs had lap times between 60 and 90 seconds. Interestingly, the provided default cars FSAE Aero and FSAE No Aero with the given track have lap times of 65.7s and 68.5s, which is higher than average IC cars (the best ones were below a minute), but it might be because of the different track, or because the default cars are not really tuned that well. So it turned out that the EV can keep up the pace with IC cars[6].

You can look at different charts, compare speeds, RPM etc. But the most interesting at this point is engine power vs. time, because from that we can calculate the actual power consumption.



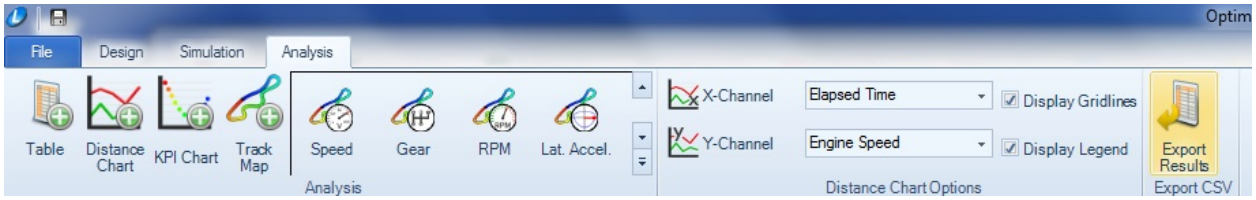
[6] at least for one round

A few comments - we see that the engine is constantly delivering power, although we are certainly braking during the lap.The engine (or rather motor in this case) still maintains RPM, because the car keeps speed, and in this particular case the speed never goes below 40km/h (~25mph) even in the tightest corner. This is OK in case of IC car, but for EV it biases the current consumption data and we will address this problem later in next section.



Analysis

Now comes the interesting part. Export data from the simulation result to CSV file (see below):



Now if you open the file, you will see a couple of sections with all vehicle and simulation settings and some statistical data (i.e. percent spent in gear 1,2,3 etc). To be able to use this file, just delete everything except simulation results section (a line with variables and lines with values). That can be then imported into MATLAB.

Current consumption

From Rinehart PM100 datasheet we know that

$$\text{Motor Voltage} = \frac{V_{DC}}{\sqrt{(2)}} \quad \text{line to line RMS}$$

and that for calculating motor power we can use the following equation:

$$\text{Motor Power} = \sqrt{(3)} * \textit{Motor Voltage} * \textit{Motor Current} * \textit{Power Factor} * \textit{Efficiency}$$

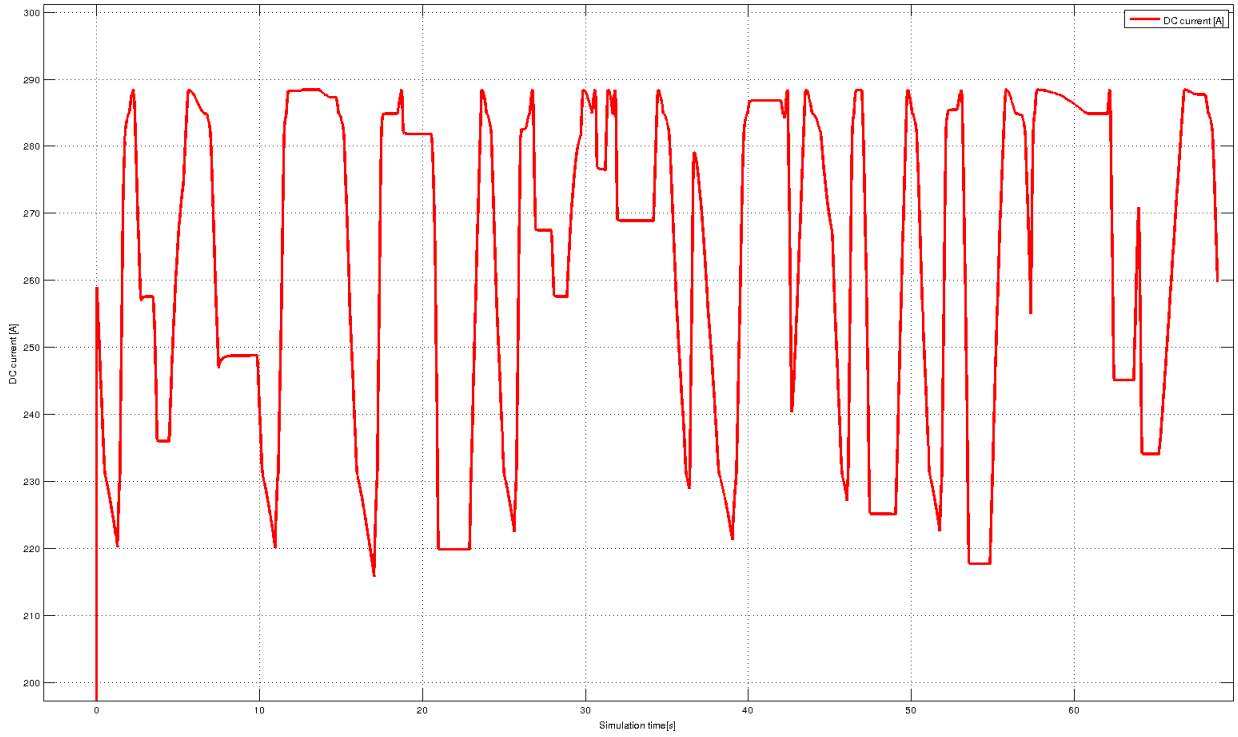
where *Motor Voltage* comes from the previous equation, *Power Factor* is .068 and *Efficiency* of the motor itself is 91.2%, and 97% for the controller efficiency, given total efficiency of 88.5%

Knowing the nominal voltage at which we operate (4x 44.4V modules gives 177.6V nominal). From the simulation we know the motor power at given time. We want to know the DC current consumption. Rearrange equation and get:

where K consists of rearranged terms f $I_{motor}(t) = motor\ power(t) * K\ [A]$

rom the motor power equation.

In our case we get the following plot:

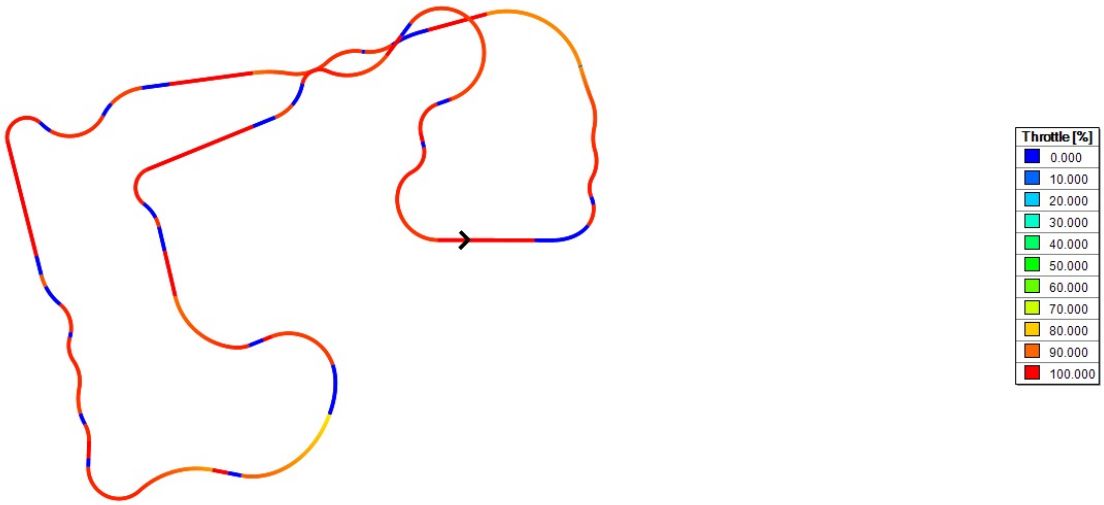


The motor current goes from 220A to 290A, with mean of 260A which is somewhat expectable given we are racing in the car, not going shopping.

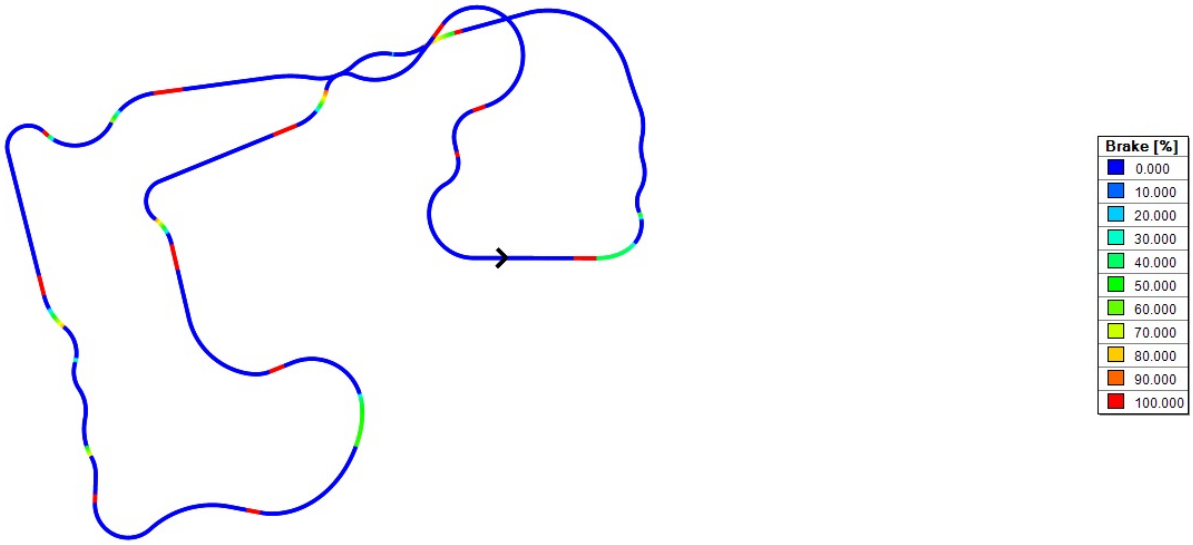
To calculate consumption, we just have to integrate the current (to get charge), i.e:

$$cons[C] = \int I_{motor}(t)\, dt$$

but that counts integrates even when the motor is braking or free-spinning so there is no current going to the motor. A simple trick - count the current only when the throttle is above certain threshold (lets say 10%). We are actually using throttle mostly either 100% or 0%, see the track map



And the same is true for braking - we almost never break, and if we break, we break hard.



The actual MATLAB code to take that in account is here:

```
for k=2:length(elapsedTime)
    dT = elapsedTime(k) - elapsedTime(k-1);
    if throttlePosition(k) > 10
        totalcur = totalcur + cur(k)*dT;
    end
    totalcur2 = totalcur2 + cur(k)*dT;
end
```

The total charge required to complete the track in 68 seconds is **15 214[C],** or more conveniently **221[A]** ($charge[C]/laptime[s]$) each second, thus **221[Ah]** (multiply by hours).

Our batteries have capacity of 32Ah, so if we drain them at rate of 221[A] they will last **32[Ah]/221[Ah]*60[min] = 8.6 [min]** which is enough to complete **7.5 laps**.

Discussion and Conclusion

There is a couple of caveats in the shown calculations, but it is a good starting point. Basically, if we could drain the batteries at the calculated rate, and could deal the motor, controller and batteries accordingly, we could keep up with the equivalent IC car for something over 7 laps on endurance (around 40% of the race).

Note that endurance is typically 22km long, the simulated track was 1.2km long, so the whole endurance would take around 18 rounds.

The batteries can provide up to 480A burst (10sec max) and 160A continuous. We never keep throttle open for more than 10 seconds, so if the batteries can provide bursts of current shortly in very short intervals we would be good here.

The capacity is another issue - adding voltage (by adding more modules) we reduce the current consumption, so the car would go further. Or we can add modules in parallel and have more Ah available, but at higher weight.

Or we will have to tamper down the max throttle (effectively the max torque of the motor), to keep current consumption down. But that compromises the speed and acceleration.

Regen? Although we are using brakes more than in regular car/bike[7], I doubt that we will ever get over 20% of regenerated battery capacity, and the actual number will be way below 10% of the capacity. So in this case something like 3Ah (roughly 50 seconds of race). So it is cool,

And indeed, we have to simulate different components to see the difference.

[7] brammo claims that in normal conditions you can regen around 5% of capacity