**LAP Simulation Documentation**

Christian Free and Brandon Strecker

December 2018

**INTRODUCTION**

This Lap Simulation was developed to evaluate different platforms (engine choices, aero sizes, etc.), perform sensitivity analysis, and observe trends in vehicle performance. It is vital to understand that the lap time values are NOT exact and it can NOT be used to size continuously variable components (i.e. a custom aero package, not an engine with a finite number of choices). However, the direction and magnitude of trends are very useful. Overall, everything we have seen suggests that it is very good at predicting vehicle performance.

The sim could be further refined ad infinitum, but at that point time would be better spent by working on modeling with a commercially available lap simulation.

**SIMULATION / CALCULATION PROCESS**

The top-level file, LapSim\_FSAEM2018.mat works with the following process:

1. Declare global variables (vehicle parameters used across multiple files)
2. Add sub-folders to path
3. Load track profiles from file

The profile is loaded as two vectors – l (length) and r (radius). Both are in meters. Each segment is either a straight or a corner. There cannot be two straights in a row, but there can be two (or more) corners in a row. Straights have radii of NaN in the r vector. **There is no racing line – everything is from the center line of the track.** The file directory also has profiles for acceleration (one straight) and one loop of skid pad (one corner).

1. Load torque curve from profile and increase resolution
2. Use Solver\_Shifting to find shift points of torque curve
3. Use Solver\_Accel and Solver\_Brake to make characteristic curves for full brake / accel

These curves show velocity and distance over time at full acceleration and full braking, considering weight transfer, aerodynamics, tire data, and the torque curve (for acceleration). The curves can be used in a straight to see the time and speed necessary to go from one speed to another.

1. Use Solver\_Corner to solve for time in all corners, assuming entry speeds can be met

This considers weight transfer, aerodynamics, tire data. This assumes that the corner is steady-state and that the entry speed can be met. This will find the quickest speed possible in all corners.

1. Use Straight\_Compiler to solve for time in all straights

This uses the characteristic braking and acceleration curves, finding the intersection (braking) point and from that, speed profile and time. It uses the corner entry and exit speeds to construct these straights. The simulation assumes that the autocross starts with a rollout at 2.2 m/s and ends with a straight ending at maximum speed.

1. Verify that straight exist speed matches corner entry speed

If this is not the case, accelerate or brake to meet the speed. **This is a weak point of the sim, since if you need to brake in a corner, it assumes that you can do this while cornering above max speed. This violates the traction circle.** This might be allowed for by momentarily going out of the center line.

1. Post-process and display final time and other plots (if requested)

**ASSUMPTIONS**

This can be most simply explained by listing out things that ARE and are NOT included.

Included:

* Vehicle parameters
  + Mass
  + Wheelbase
  + Track width
  + CG height and longitudinal location
  + Tire diameter
* Drivetrain Parameters
  + Transmission, primary, and final drive ratios
  + Engine, wheel, and driveline inertias
  + Shift time
  + Transmission efficiency
* Aerodynamic Parameters
  + Air density
  + Frontal area
  + Lift and drag coefficients
* Weight transfer
* Engine torque

Not included:

* A racing line – the line is always in the center of the track
* Driver error or drivability – assume perfect ability to maximize grip
* Any transient effects – everything from tire transients to corner entry yaw moment
* Polar moment – this is more of a transient thing
* Compliance – difficult to measure, predict, and analyze
* Center of pressure – assumed to be at same height as CG, center plane, and 50% wheelbase (downforce evenly applied over 4 wheels)
* Drag moment is not included in weight transfer
* Pitch, roll, and yaw sensitivity of aero

**RESULTS AND VALIDATION**

Compared to actual lap times, the simulation is slower in autocross but faster in skid pad and acceleration. This has not been completely verified, but it is believed that lower autocross performance is due to lack of a racing line. Higher acceleration and skid pad performance comes from ideal drivers (and ABS), which perfectly perform clutch actuation, shifting, slip management, reaction time, etc.

One important component related to accuracy is tire data tuning. Constant factors are applied to data sets from TTC (Tire Testing Consortium) to match peak longitudinal and lateral values found from on-car data.

More work is necessary to validate the simulation data, against both car data and results from commercially available simulation packages. But so far, no discrepancies or oddities have been found and left “broken” in the simulation files.

**FILE STRUCTURE**The top-level folder consists of two sub-folders: one for documentation and one for the simulation files.

The simulation top-level folder consists of two MATLAB functions and four folders. Two of the folders are for 1D and DOE analysis and will be explained later. The other two folders are sub-folders used to organize helper files for the simulation.

The MATLAB files in the simulation top-level folder are the main file, LapSim\_FSAEM2018.m, and the file used to declare parameters, LapSim\_Declarations.m.

The Data Storage and Handling folder has data files for the torque curve, track files, and MATLAB functions for the tire data. The Solvers and Compilers folder has solvers for acceleration, braking, cornering, and shifting, and a compiler for straights. There is also a function from the MATLAB file exchange called intersections.m used to find the intersection between the characteristic acceleration and braking curves (braking point).

**ANALYSIS PREP FILES**

**SOLVER AND COMPILER DETAILS – Solver\_Accel**

**SOLVER AND COMPILER DETAILS – Solver\_Brake**

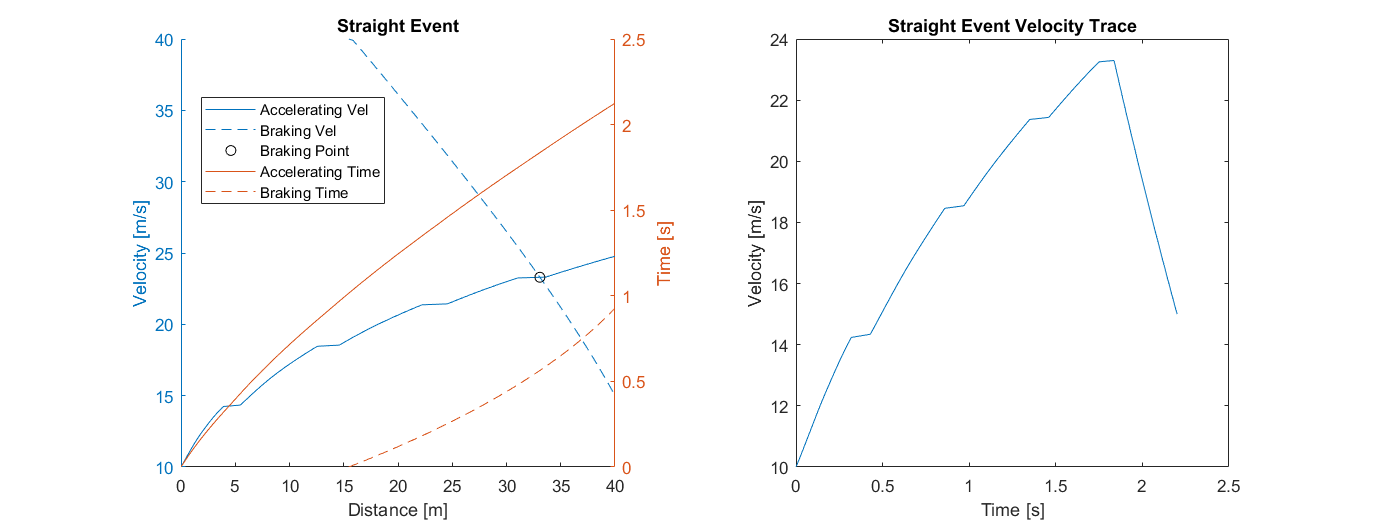
**SOLVER AND COMPILER DETAILS – Solver\_Corner**

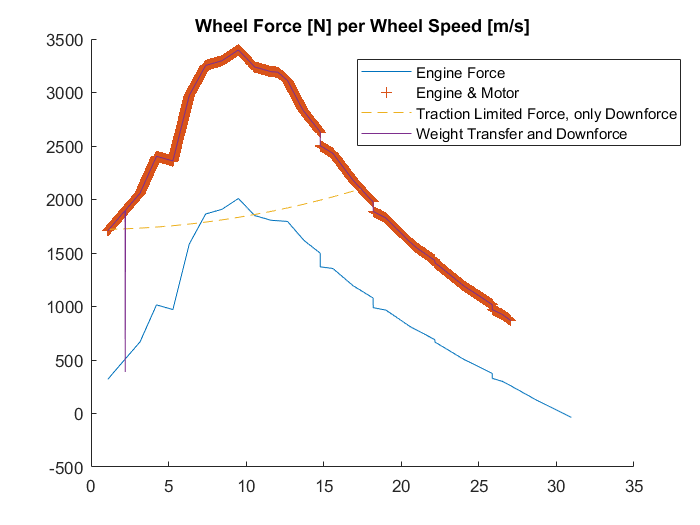
**SOLVER AND COMPILER DETAILS – Solver\_Shifting**

**SOLVER AND COMPILER DETAILS – Straight\_Compiler**

**MATLAB LAP SIMULATION**

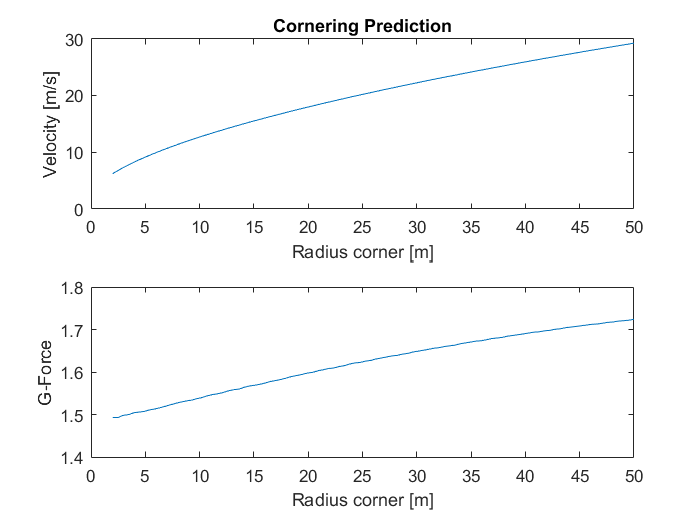
The MATLAB Lap Simulation was constructed using numerical methods and vehicle dynamics equations to evaluate a racing vehicle’s sensitivity to component changes. Its inputs are the Michigan 2018 FSAE autocross track, simplified to a series of constant radius turns and straights of various lengths, and a conglomerate of powertrain and vehicle parameters. It simulates longitudinal vehicle dynamics by creating velocity traces of acceleration and braking events and then overlaying them on a track straight to compute the braking point and time across the straight. Weight transfer is assumed to be instantaneous and a tire loading model is used to allow for the additional grip of downforce and weight transfer. Figure 5 corresponds to a representative straight with an entry velocity of 10 m/s and an exit velocity of 15 m/s. The plateaus in accelerating velocity are points where the car shifts gears and cannot accelerate for 0.1 seconds (the configurable time it takes to shift). Figure 6 is a graph of tractive force versus wheel speed for the proposed hybrid vehicle. Note the significant increase in tractive effort with a motor and the lack of traction when weight transfer is neglected.

**Figure 5:**(Left) A composite graph of the straight event showing velocity under constant acceleration, velocity under constant braking, and the braking point at which the driver should stop accelerating and start braking. (Right) The resultant velocity trace from the straight event



**Figure 6:** Tractive effort graph with the purple line highlighted in red showing the resultant trace

Corners in the MATLAB Lap Simulation are computed by assuming steady state weight transfer. Transitional vehicle dynamics as well as angular dynamics are neglected. Key variables are vehicle weight, tire radius, center of gravity (CG) height, CG placement, and coefficients of drag and downforce. A tuning variable is multiplied to the tire model to compensate for suspension effects and match the cornering accelerations seen on the actual GT FSAE vehicle. A chart of expected cornering speeds and accelerations for car #48 are shown below in Figure 7.



**Figure 7:** Steady-State Cornering Prediction

**ACKNOWLEDGEMENTS AND BACKGROUND**

A very first iteration of what could be called a vehicle simulation in MATLAB was built by Christian Free (F15–F19 PT member, F18 PT lead) with the help of Elliot Blotcha (F15–F18 CT member, essentially F18 CT lead) and Tyler McDaniel (F16–F18 driveline leader). It was just a script with a few helper functions to calculate a time for the acceleration event, based on drivetrain ratios and efficiencies, a torque curve input, and longitudinal vehicle dynamics. The tire model used only had a constant tire force limit, not varying with load. This project was developed and built in the F18 cycle.

A separate simulation was built in the summer between F18 and F19 by Dustin Roth (F16 SU member, F17-F18 SU lead), since he had received feedback from design judges that it would be useful for us to build our own simulations. The sim considered longitudinal and lateral vehicle dynamics, with tire data directly from TTC (Tire Testing Consortium). This data is unrealistic (mu too high) since the tires are tested on a sand belt. The sim worked by discretizing the autocross track into straights in corners (as with the final sim) and assuming the driver and powertrain could maximize the grip of the tires in all sectors. As such, there was no torque cure input – the assumption was that performance was always limited by the tires.

The final Lap Simulation was compiled, developed, and built by Christian in the first half of F18. The project was undertaken for several reasons – evaluating options for his senior design (aero foil muffler), analysis and component selection for ME 4013 projects (Hybrid Vehicle Powertrains), general help for the team in platform selection, and an analysis tool for a future A-Mod car. Essentially, it combined code from both previous projects (powertrain dynamics from acceleration sim, vehicle dynamics and tire model from previous autocross sim) and used them to more accurately predict performance. Part of this involved applying scaling factors to TTC data to match peak acceleration values found on-car.

This documentation and small bits of formatting were done by Christian and Brandon Strecker (F16-F18 PT member, F19 PT co-lead).