Vehicle Dynamics

Week 6 - Laptime Simulation

Oliver Rose

University of Strathclyde Motorsport

28/10/2025

What is Laptime Simulation?

A Laptime Simulator (or **lapsim**) is a computer model of a vehicle that predicts how quickly the vehicle can complete a course.

Lapsims can vary in complexity, from simple timestepping calculations in Excel through to complicated transient models or driver-in-the-loop simulations.

As a model becomes more complex, it becomes more prone to error and harder to validate. You should always use the simplest possible model that accounts for all the behaviour you want to study.

Vehicle Models

Point Mass: Vehicle is treated as a concentrated point mass. Useful for analysing basic sensitivities.

Bicycle Model: Vehicle is represented by a front and rear axle. Longitudinal load transfer occurs when braking and accelerating.

Four Corner: All four tyres of the vehicle are represented. Both lateral and longitudinal load transfer are considered.

Four Corner with Suspension: Effects of suspension geometry (camber, toe, etc.) are considered. Useful for optimising vehicle setup.

Quasi-Steady-State (QSS) Simulation: Acceleration is constant in each section of the track mesh. Simple to solve, but unable to account for transient effects such as yaw acceleration or cell temperatures.

Transient Simulation: Transient effects like temperature are modelled. Much more complicated to solve than QSS, but able to explore more complex vehicle behaviour.

Driver-in-the-Loop (DITL) Simulation: The vehicle is simulated in real-time, with control inputs provided by the driver. Able to explore the 'driveability' of a vehicle, but also subject to driver error.

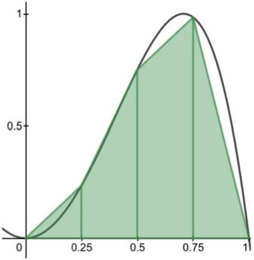
Time Basis

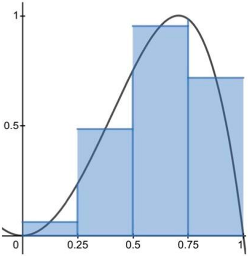
- Each step takes a fixed amount of time
- Distance calculated with $s = ut + \frac{1}{2}at^2$
- Good at simulating the acceleration event

Distance Basis

- Each step is a fixed distance
- Time calculated with $t = s/v_{avg}$
- Good at simulating autocross and endurance events

We can use numerical integration methods like the trapezoidal or the midpoint rule to improve accuracy.





Existing Laptime Simulations

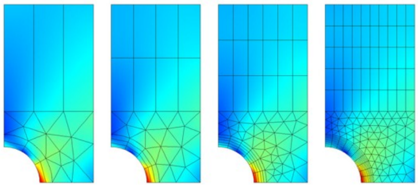
OpenLAP: Open-source point mass laptime simulation written in MATLAB. Useful starting point, but vehicle model is very limited.

Bailey and Murray's Lapsim: Four corner laptime simulation written in MATLAB. Much better vehicle model with suspension geometry and an electric powertrain, and with a more accurate tyre model.

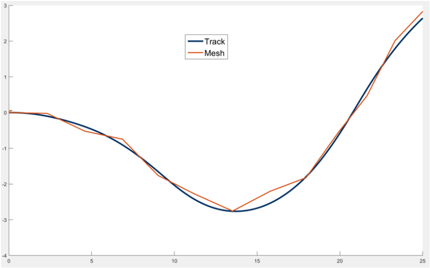
USMLap: Laptime simulation written in Python. Using Murray's lapsim as a starting point, optimised to allow it to run much faster. Will include a transient solver, allowing for accumulator state-of-charge and cell temperatures to be accounted for.

Quasi-Steady-State Simulation

- The track is broken down into a mesh of discrete time or distance steps (similar to finite element analysis)
- Acceleration is assumed to be constant at each step
- The finer the mesh, the more accurate the result

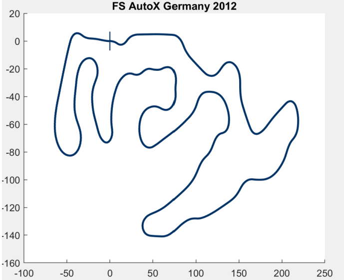


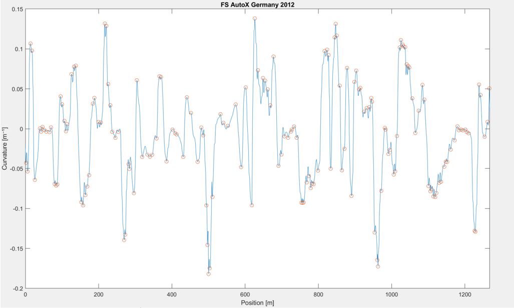
Finite Element Mesh



Track Mesh

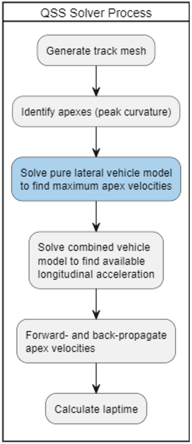
The shape, elevation, banking, and grip factor of the track is defined in a spreadsheet. This is converted to a mesh with approximately equal distance steps.

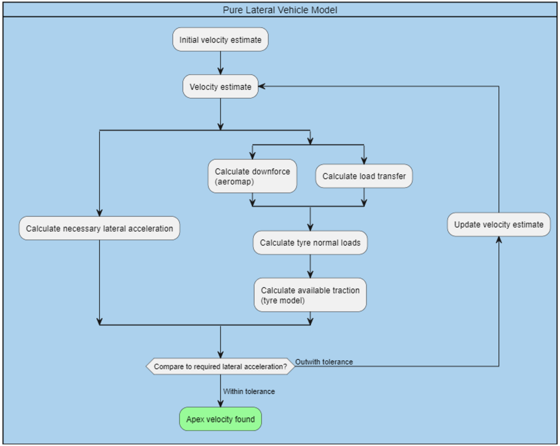




Apexes are identified as the points where curvature is at a maximum.

Aero forces and load transfer are affected by velocity and acceleration. Therefore, an iterative process is required.





Without validating our simulation, we must be careful what conclusions we draw from it.

We can observe **trends** (e.g. laptime decreases as mass decreases).
We cannot use exact **numerical** values (e.g. decreasing mass by 1kg decreases laptime by 0.1s).

How can we validate our simulation?

- Compare to real test data from the car. This requires testing time and data acquisition.
- Compare to test data from other sources. Not as strong as using data from our car. However, this can be used to show that the lapsim functions correctly.

Correlation

What can we do if our model doesn't match experimental data?

The most important thing is to understand **why**. Compare various testing data channels to simulation data channels to diagnose where the problem is occuring.

How do we bring our model in line with the data?

- Fix any errors in the model
- Use a more complicated physics model
- Add **correlation coefficients**

If correlation coefficients are used, they should be as simple as possible, and their use should be fully justified (e.g. scaling coefficient applied to tyre grip to account for tyre degradation).

USM