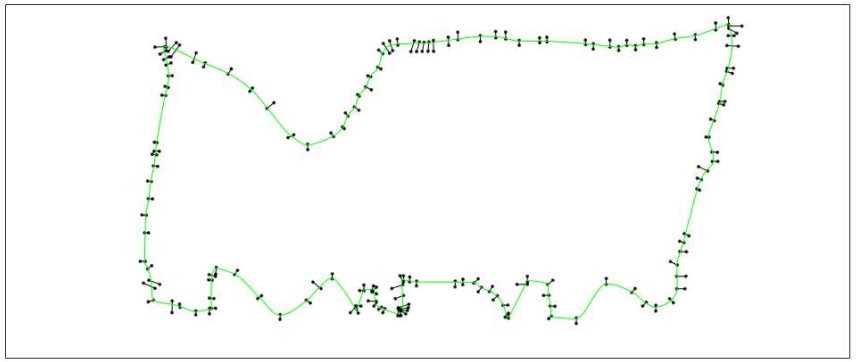
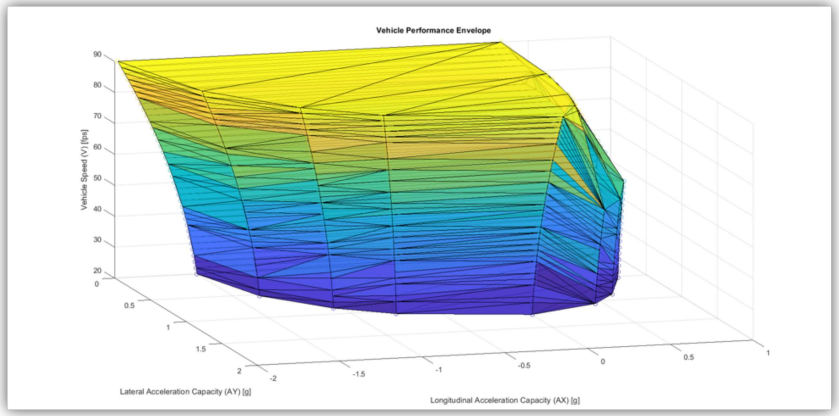
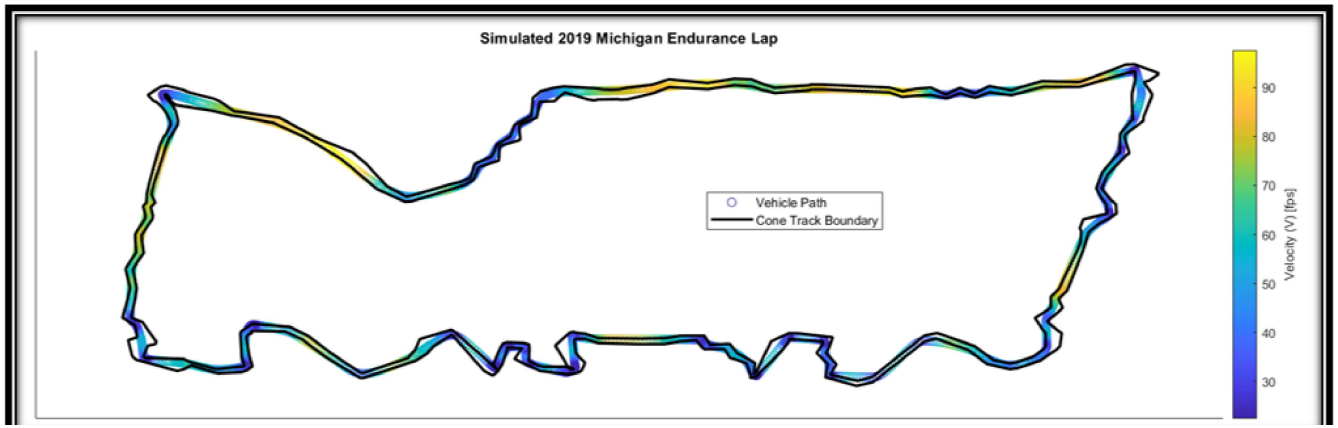
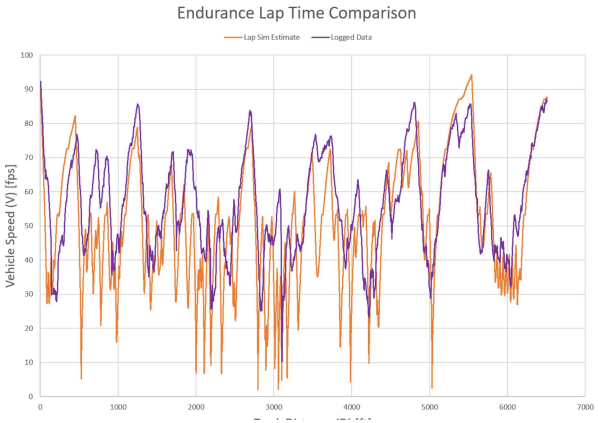
Lap Time Simulation

Please enjoy the 2-minute summary gallery below, scroll further to explore my work in greater detail, or explore my code in this GitHub repository.

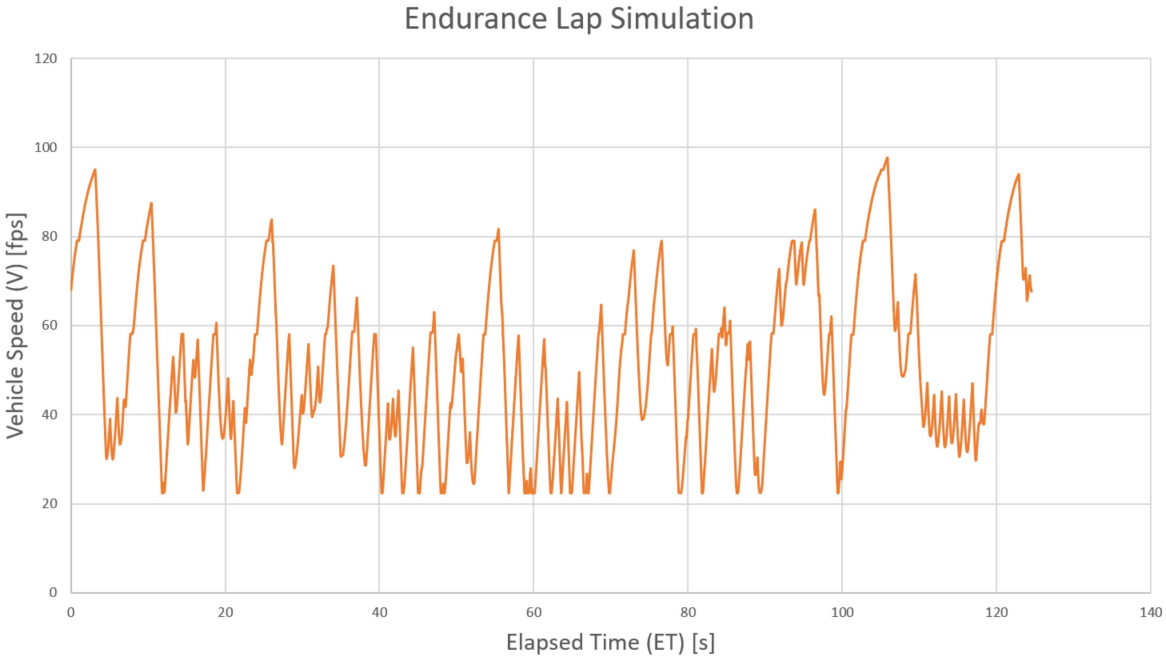








Lap time simulation can be an invaluable tool for developing a vehicle. The ability to predict the performance of various vehicle configurations can help guide the engineering design process and ensure that resources are being spent in the areas that will reap the highest reward. On this page, I will discuss the process I used to develop my lap time simulation code, and explain the key decisions I made to achieve my target attributes.



Example output plot of a simulated run on the FSAE Michigan 2019 Endurance Map

Goals and Objectives

The first thing to discuss is what utility you are trying to get out of a lap time simulation, as this drives the major development decisions and what features get included. For my FSAE team, the benefit of a lap sim does not come from being able to perfectly recreate a real world lap time. Rather, the focus is in being able to capture changes in relative performance instead of absolute. As long as the representation of vehicle performance is accurate enough to capture the effect of changes such as reducing weight or improving aerodynamic efficiency, then a lap sim can be used to compare concepts and identify areas of development where we can get the most bang for our buck.

With that in mind, these are the primary objectives identified at the beginning of the project:

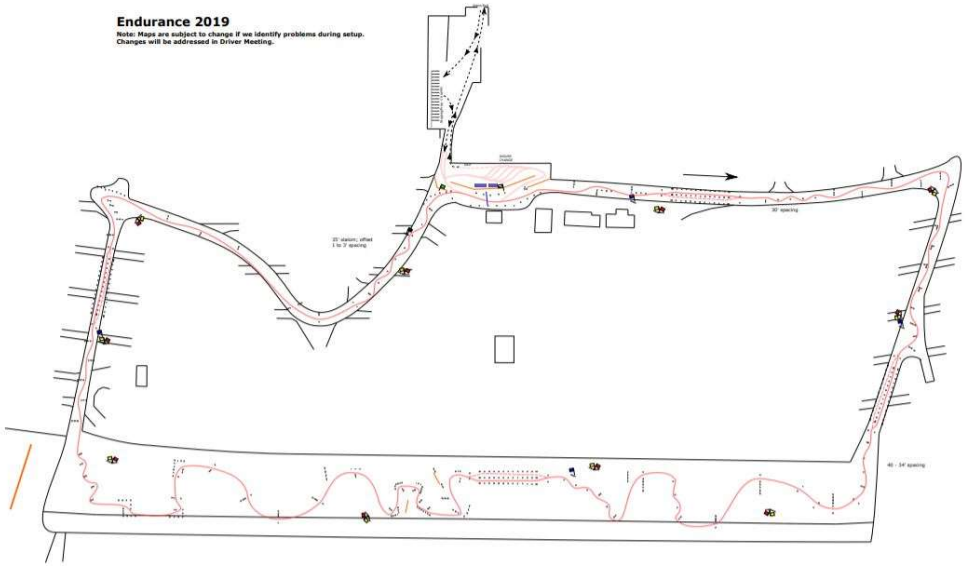
Lap Time Simulation Development Criteria	
Objective	**Justification**
Evaluate vehicle trajectory based on vehicle size and performance characteristics	Confining vehicles to a set path can artificially benefit certain vehicle concepts over others. For example, a narrower track width vehicle may have reduced cornering capacity from the increase in lateral load transfer, but that could be outweighed by the smoother line that can be taken through slaloms and other tight corners
Incorporate modularity in vehicle performance modelling	Keeping sustainability in mind, it is important to keep this tool relevant even as other design and modelling tools improve on the team. This lap sim should be able to incorporate vehicle models of increased complexity with minimal implementation effort.
Focus on computational efficiency with an acceptable level of accuracy	As long as a proper threshold of accuracy is met, there is greater value to be gained from being able to explore a wider design space in a shorter timeline. This was qualitatively evaluated, as I did not want an arbitrary constraint to prematurely stunt the development process
Be able to capture the effect of key subsystem interactions on performance	One of the long term goals of this simulation is to serve as a foundation to maximize the development of the vehicle as a collection of systems working together. If a change in one subsystem affects the performance of another, we want that change to be revealed in simulation results

Vehicle Path Generation

The first step is to be able to chart a feasible and realistic vehicle trajectory for the lap sim to evaluate. My goal was to re-

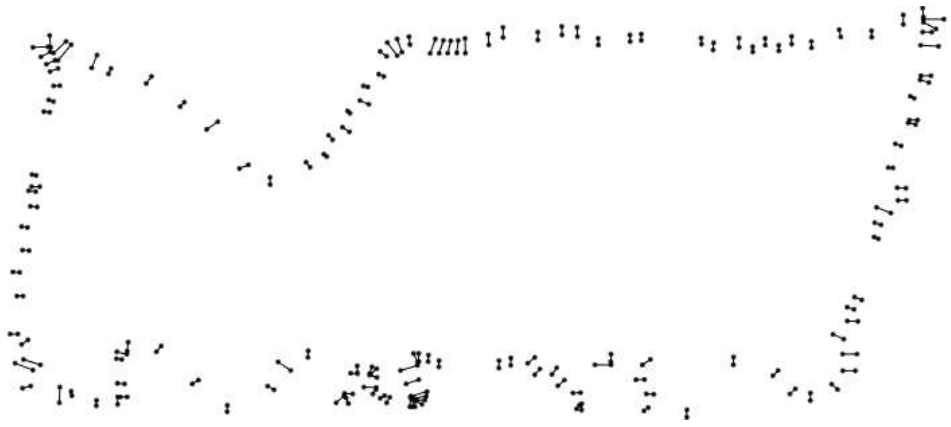
create the 2019 FSAE Michigan Endurance and Autocross courses. These would be the closest representations of potential future courses that our FSAE cars would run, making them the ideal starting point for performance evaluation. As an added bonus, predicting lap time and comparing to the 2019 score sheets enables us to create points scoring predictions as a means of comparing different concepts.

This is an example of a track map provided by FSAE. There are some cones represented by black dots and a scaling legend, but it is a far cry away from a well defined vehicle path. First, I imported images like these into Solidworks and created a sketch over the image, placing dots over each defined gate cone. I also added more gates to more sharply define the intended course trajectory. I was then able to import the coordinates into an Excel spreadsheet, and properly scale them to match the distance units of interest.



Source: "Endurance 2019." Fsaeonline.com, Society of Automotive Engineers, Apr. 2019, www.fsaeonline.com/cdsweb/gen/DownloadDocument.aspx?DocumentID=73d7e97f-b223-4a8e-af7d-b7143b5afec2.

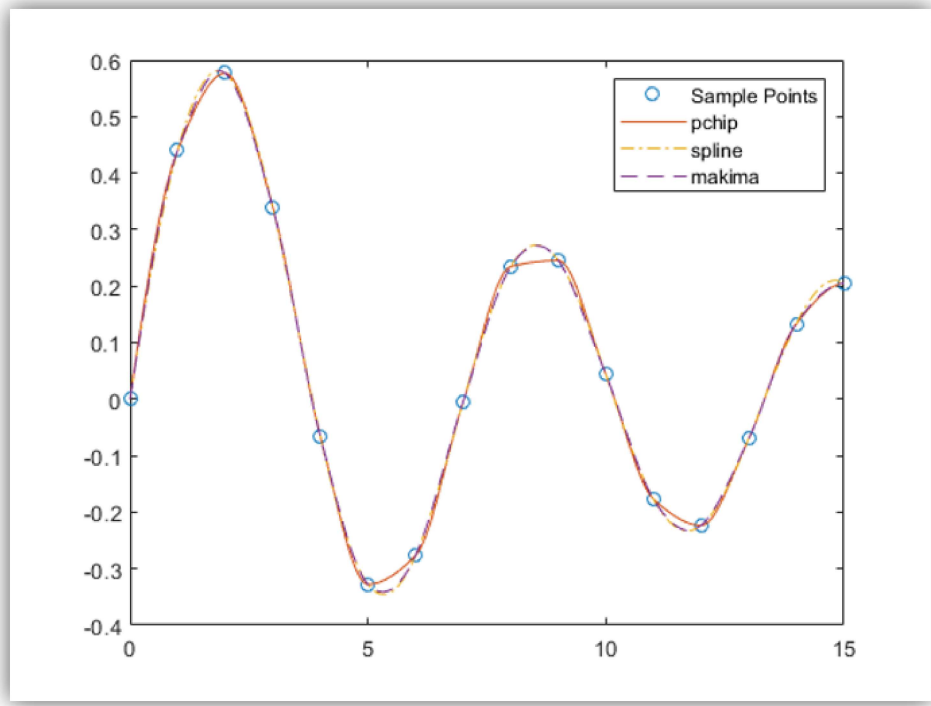
Once the points were in excel, I brought them into MATLAB



Endurance track represented by gates in MATLAB

Once the points were in excel, I brought them into MATLAB to replicate the track map. At this point, there were enough gates to create a vehicle trajectory. This was defined by a vector the same length as the number of gates, with values between zero and one. Zero meant that the vehicle crossed a given gate right at the very edge of the inside cone, and one meant the vehicle crossed at the absolute outside, and any value in between was an interpolation between the two. These points could be connected to create a fast and distinct definition of vehicle path. On top of that, upper and lower bounds can be automatically updated with vehicle size, allowing a difference in dimensions to affect the final trajectory and helping meet one of the core objectives.

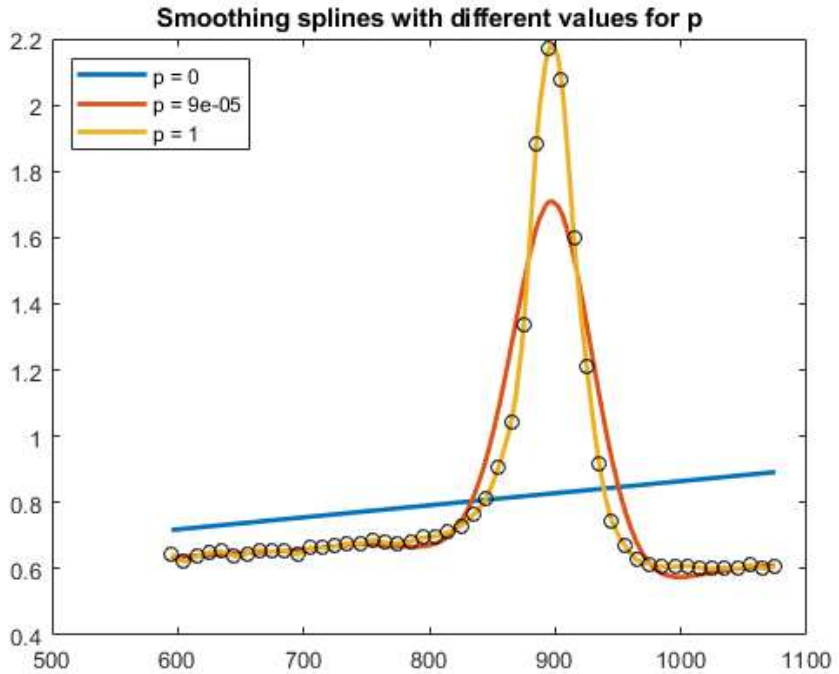
This brought a new challenge of finding the best way to connect those dots. MATLAB has a variety of different spline fitting algorithms available, which can create completely different curves from the same sample of points.



Example comparison of available MATLAB Spline fit functions

Source: MathWorks Help Center, MathWorks, 2006,

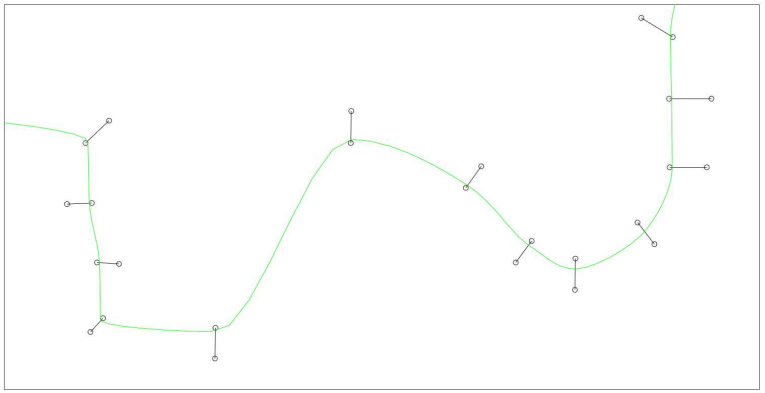
www.mathworks.com/help/matlab/ref/spline.html.



Source: "Smoothing Splines with Different Values for p." MathWorks Help Center, MathWorks, 2006, www.mathworks.com/help/curvefit/csaps.html.

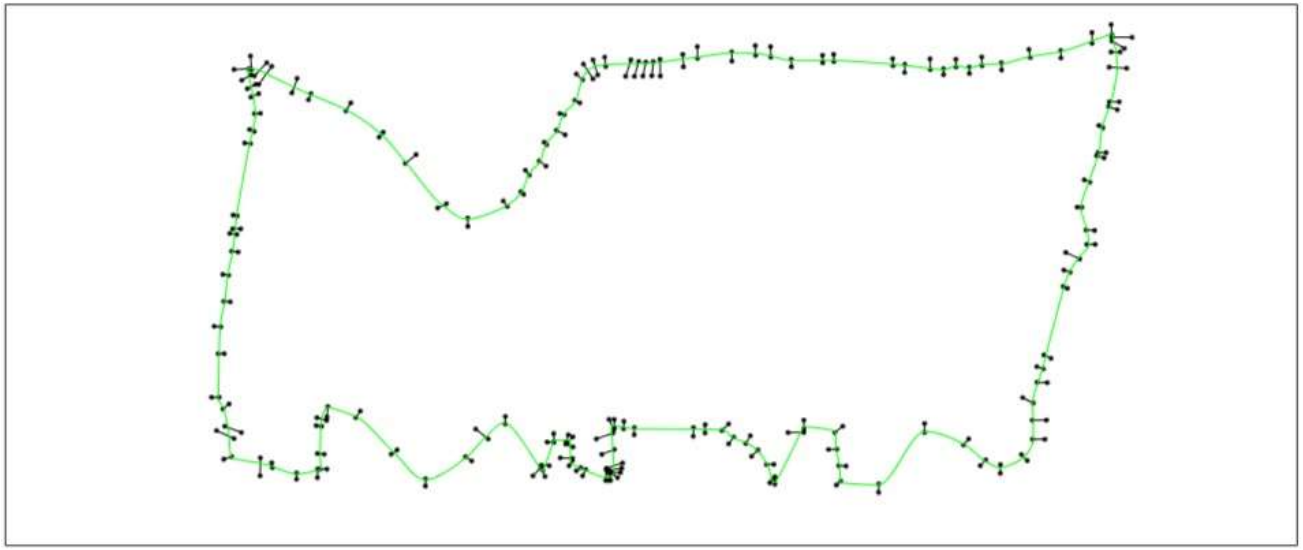
On top of that, there are a variety of fit parameters, such as the smoothing parameter, that could be adjusted to fine tune the fit. Ultimately finding the best combination was an iterative guess-and-check approach until I found the combination that yielded the most organic and realistic vehicle trajectories. I did this with the help and input of the team drivers, who had much more insight on what the racing line should look like.

The final configuration selected uses the "Pchip" spline fit. Other spline curves had smoother overall curvatures, but were very sensitive to inconsistency in tight areas on the track. Makima, on the other hand, was able to create consistent and repeatable trajectories across many track configurations. The biggest trade-off is the occasional sharp break in curvature. It is important to remember here that no trajectory is going to be perfect, but after review with the drivers it was decided that this method was representational enough to move forward.

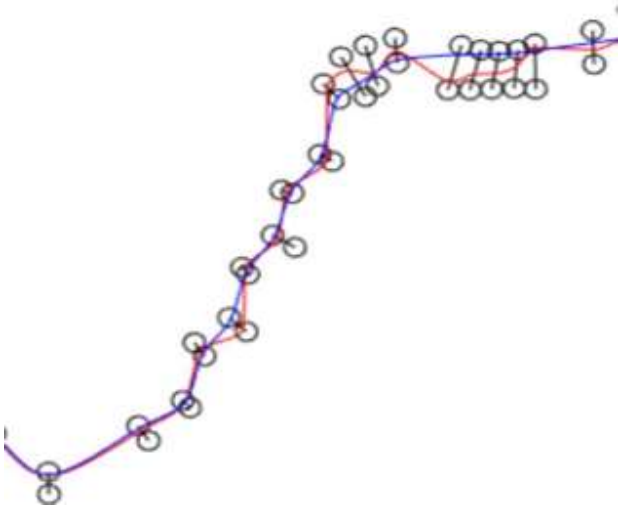


Sample of a vehicle autocross trajectory generated using Pchip

The next question is, how do you find the optimal line? It is well accepted that often times, the shortest path through the track is not necessarily the fastest path. However, it makes a good starting spot. To begin with, MATLAB's fmincon function was used to optimize the trajectory to minimize distance traveled, with the constraint that the instantaneous curvature should not exceed the minimum FSAE turn radius at any point along the track. Once the lap sim was operational, it was used as an initial point to optimize the racing line and minimize time elapsed instead of distance traveled.



Example of optimized endurance racing line



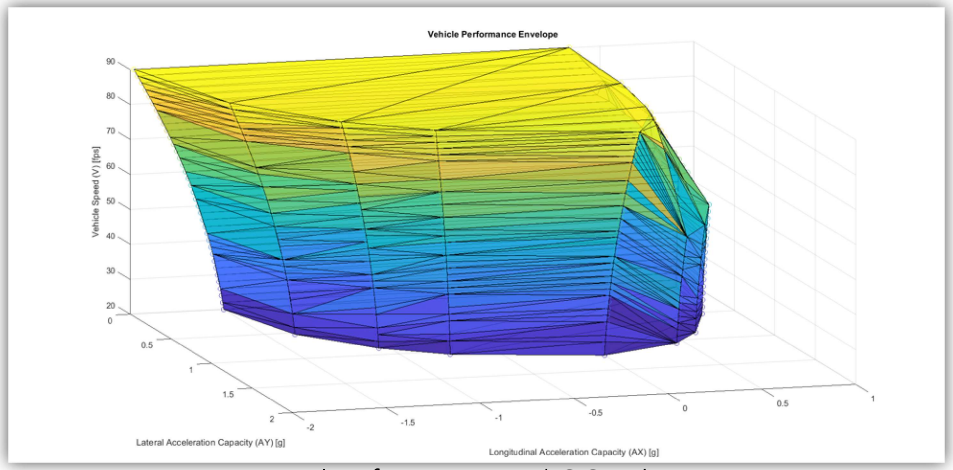
Close up between randomly generated (red) trajectory vs optimized (blue) trajectory

Vehicle Performance Modelling

Selecting a vehicle performance modelling method is a critical design decision. Painting in broad brush strokes, it comes down to striking a balance between accuracy of the lap and the computational cost of achieving that accuracy. It is important that lap simulation results reflect realistic vehicle behavior, but it is equally important that a single run doesn't take 48 hours, which would limit opportunities for design space exploration and optimization.

I decided to evaluate vehicle performance in a quasi-steady state approximation. This assumes that the vehicle is simply moving between states of equilibrium at a rate that is slow enough to ignore transient effects. In reality, this is not true. A vehicle is almost always in a transient state, with key parameters such as damping and rotational inertia affecting time domain response. However, transient simulations are often so complex that analytical solutions are impossible, and numerical solutions can be prohibitively expensive in computation time. Using the right size time step, quasi-steady state simulations can maintain a relatively high degree of accuracy for a small fraction of the time cost.

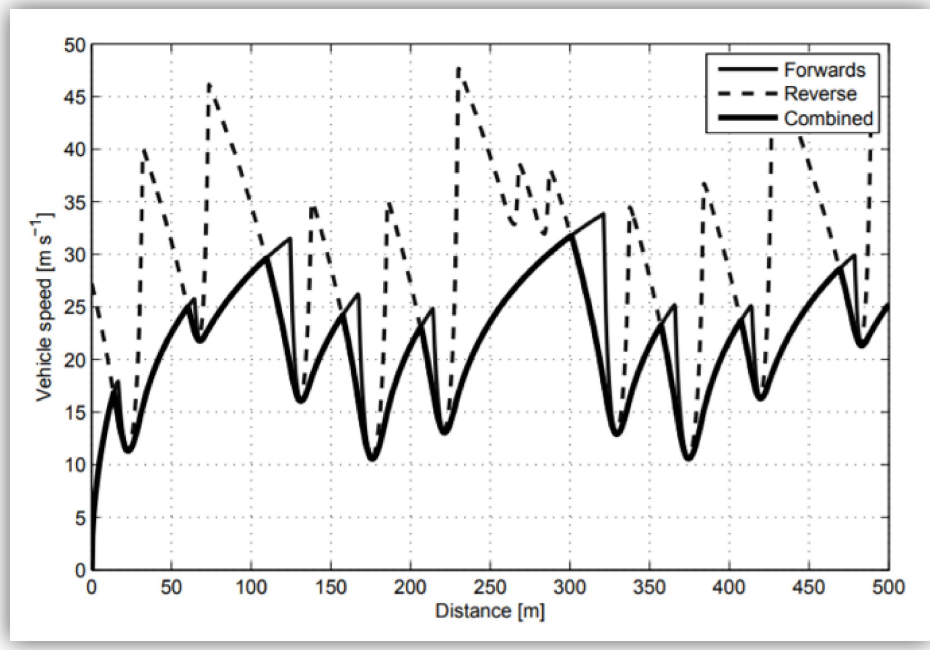
Quasi-steady state simulations require knowing the steady state capabilities of the vehicle in any given operating condition. This was achieved using a GGV diagram. This is essentially a G-G friction ellipse expanded into 3-D with velocity as the third axis. Each point along the diagram represents a steady state performance condition, and the model discreetly moves from point to point across the diagram.



Example of a generated GGV diagram

The implementation of the GGV diagram is the most significant feature of this lap simulation, because it achieves the goals of fast evaluation and modular construction. The lap simulation blindly takes a GGV model input, blind to how that model was generated. This means that different models of varying complexity and scope are interchangeable in the program, so

long as the model creates a GGV diagram. This also helps the simulation solve significantly faster, as it eliminates the need to evaluate the performance capabilities of the vehicle in real time as it moves across the track.

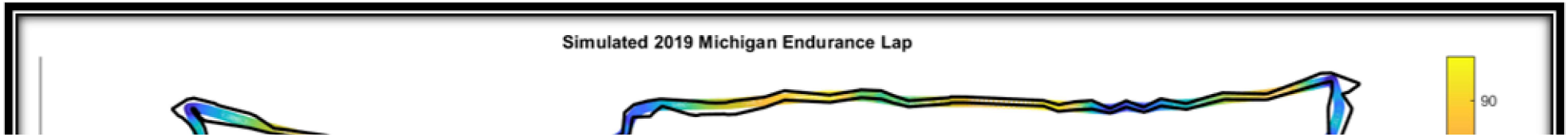


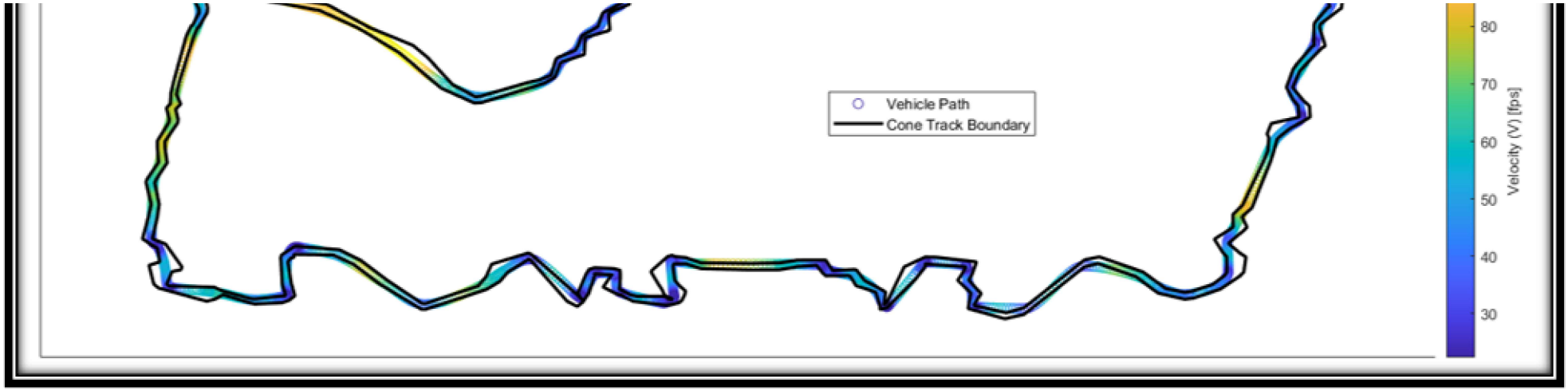
Visualization of the forward/reverse method

Source: Criens, C. H. A., Dam, ten, T., Luijten, H. J. C., & Rutjes, T. (2006). Building a MATLAB based Formula Student simulator. (DCT rapporten; Vol. 2006.069). Eindhoven: Technische Universiteit Eindhoven.

The final vehicle states across the track, and subsequent performance, were evaluated using the forward/reverse method, which is visualized on the left. In essence, the vehicle moves forward across the track, and accelerates whenever possible. If it reaches a point of curvature that it is travelling too fast to navigate, the speed immediately drops to the maximum possible. Then, the same process is repeated in reverse, and the final vehicle performance at any point in the track is the minimum of the two. This is significantly more efficient than incorporating complex logic protocols to anticipate corners and braking points.

When all is said and done, this is what the end result looks like!

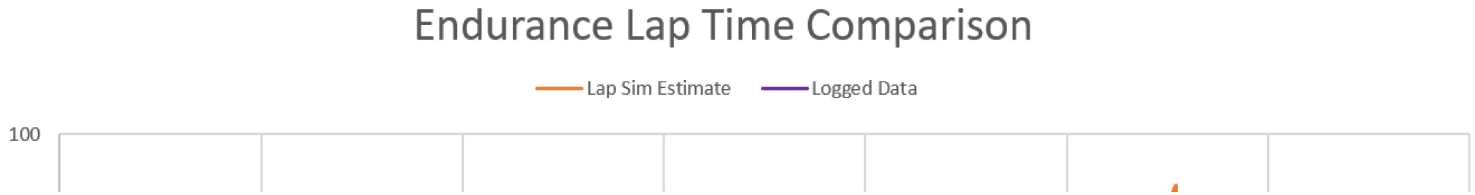


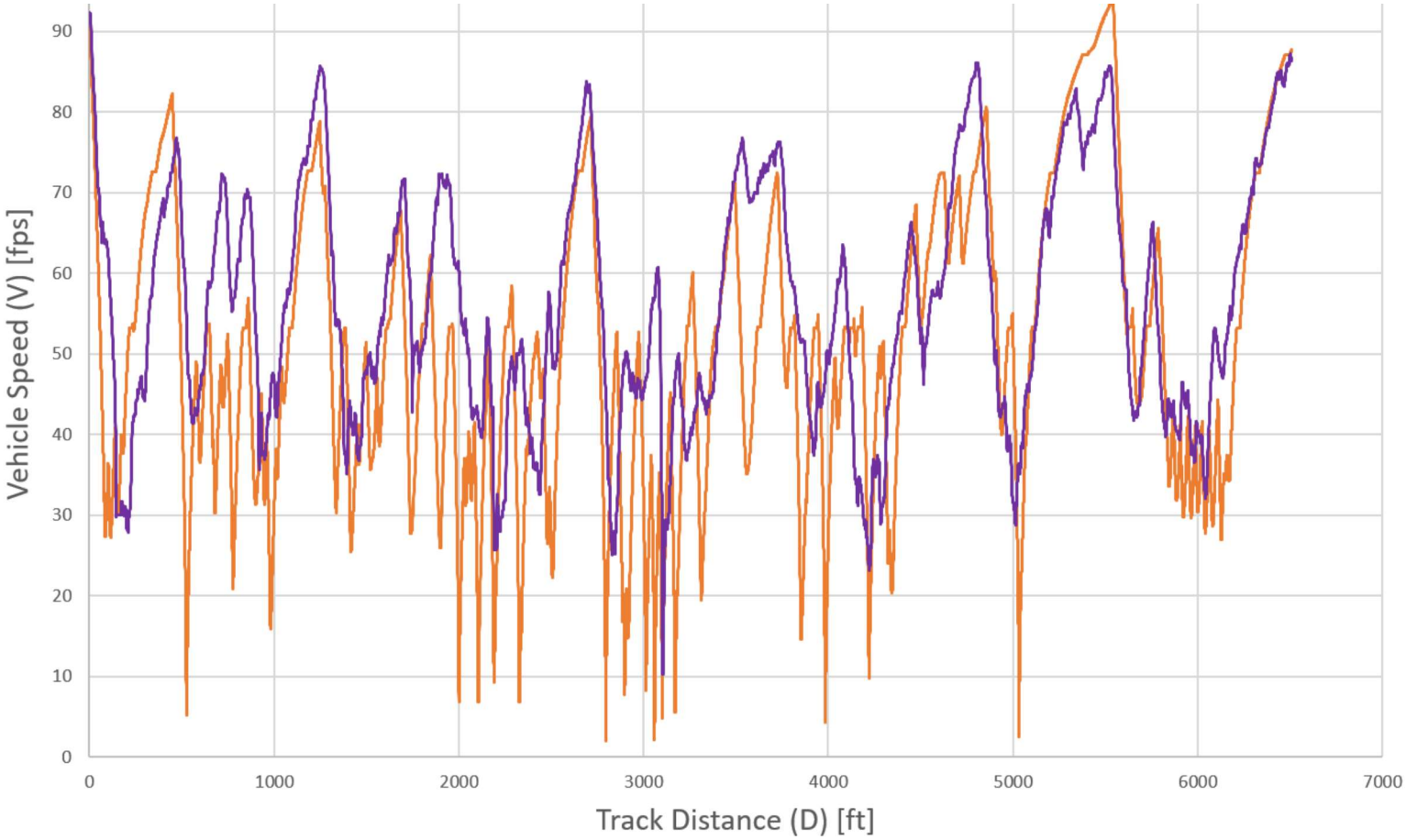


Results and Correlation

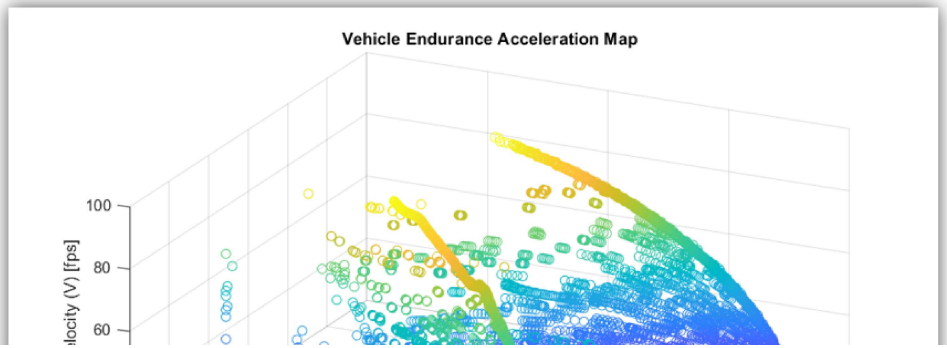
The most important question is, how well does it work?? Does it accurately represent on-track vehicle performance? Some insight can be gained from the comparison between simulation and logged data below for the 2019 FSAE Michigan Endurance track. The simulation is a representation of Tiger20, the car with which we competed that year, and the logged data comes from our fastest recorded endurance lap.

* Disclaimer: When the final simulation was evaluated, the vehicle distance traveled was about 2-3% off of the published track distance, so the x axis was re-scaled to make the distances match.



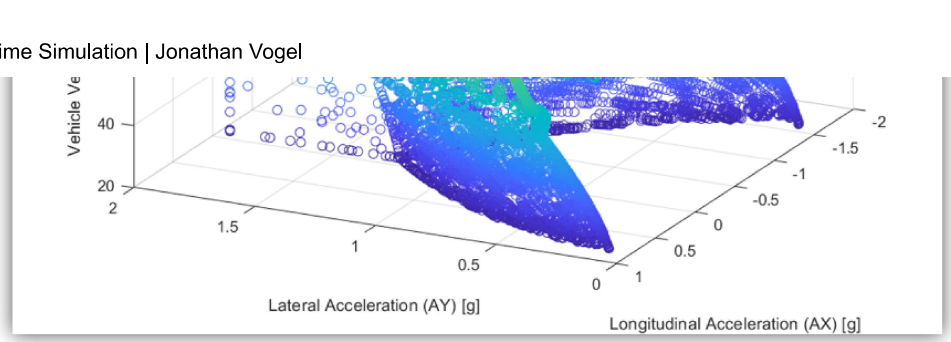


What is first apparent is that the correlation is nowhere near perfect. There are a few key differences. For example, the lap simulation drops to much more dramatic minimum velocities. This is attributable to the Makima trajectory that was generated, which featured unrealistic sharp changes in curvature that created local minima in the instantaneous turn radius. This correlates to the point cloud on the right, where the outlier lateral acceleration points correspond to the points when the vehicle entered these outlier points at a



proportionally high speed. In addition, the lap sim features much more jagged velocity traces than the logged data. This is due to the nature of the quasi-steady state evaluation, representing an idealized optimal performance where the vehicle can extract the maximum acceleration and braking for a tight sequence of corners where a human driver would

instead maintain a more consistent speed. All of this being said, the general trends in both graphs, as well as macro performance statistics, match quite well.



Note the area to the far left, where the points fall outside of what is otherwise a neat construction of the GGV diagram from actual lap simulation outputs.

The table below is a testament to the possibilities from using a GGV diagram. The statistics represent the same vehicle configuration navigating the same course, with varying levels of detail and complexity. The orange model is an elementary two-track model, while the blue and green incorporate pitch, roll, and different level of suspension kinematics. As the models increase in detail, the computation time increases. But the similarity in results speaks to the feasibility of using a GGV diagram as a consistent tool for evaluating vehicle performance. This demonstrates the true power of this lap sim as a flexible design tool, which is the ability to pick and choose exactly how much detail you want to include and how much computational efficiency you are willing to give up, without making any changes to the lap simulation code itself.

	2019 Endurance Lap Sim Insights					
	Standard Model		Camber Model		Camber + KPI Model	
Parameter	Value	Units	Value	Units	Value	Units
Maximum Speed	64.7	mph	65.2	mph	65.3	mph
Average Speed	35.9	mph	35.3	mph	35.4	mph
Maximum Lateral Acceleration	1.473	g	1.44	g	1.438	g
Average Lateral Acceleration	0.97	g	1.03	g	1.03	g
Maximum Longitudinal Acceleration	1.12	g	1.01	g	1.01	g
Minimum Longitudinal Acceleration	1.94	g	1.91	g	1.92	g
Average Acceleration	0.69	g	0.69	g	0.69	g
Average Decceleration	1.57	g	1.56	g	1.56	g
Amount of time spent cornering	54.22	%	54.64	%	54.63	%
Amount of time spent accelerating	31.35	%	31.05	%	31.08	%
Amount of time spent cornering	14.43	%	14.31	%	14.28	%

Comparison of lap sim results from different vehicle models with varying levels of complexity

The logged data comparison demonstrates clearly how a quasi-steady state lap simulation with the simplifying assumptions discussed in this page will not capture the true behavior of the vehicle on track. However, comparing the

general trends of vehicle performance, the case is also presented that this model is accurate enough to act as a comparison tool to evaluate the performance deltas across various vehicle configurations. Included below are the results from some sensitivity studies used to "gut-check" the efficacy of the program of capturing these performance interactions. To create these results, each 2019 Michigan dynamic event track was captured and evaluated, in order to predict the total dynamic event points scored as an ultimate comparison.

Skidpad Score		Drag @ 20 m/s [lbs]							
Absolute		60	80	100	120	140	160	180	200
Downforce @ 20 m/s [lbs]	100	57.70097	57.70097	57.70097	57.70097	57.70097	57.70097	57.70097	57.70097
	120	58.45761	58.45761	58.45761	58.45761	58.45761	58.45761	58.45761	58.45761
	140	59.24308	59.24308	59.24308	59.24308	59.24308	59.24308	59.24308	59.24308
	160	60.01129	60.01129	60.01129	60.01129	60.01129	60.01129	60.01129	60.01129
	180	60.81544	60.81544	60.81544	60.81544	60.81544	60.81544	60.81544	60.81544
	200	61.60516	61.60516	61.60516	61.60516	61.60516	61.60516	61.60516	61.60516
	220	62.42065	62.42065	62.42065	62.42065	62.42065	62.42065	62.42065	62.42065
	240	63.22741	63.22741	63.22741	63.22741	63.22741	63.22741	63.22741	63.22741
	260	64.04267	64.04267	64.04267	64.04267	64.04267	64.04267	64.04267	64.04267
	280	64.90677	64.90677	64.90677	64.90677	64.90677	64.90677	64.90677	64.90677
	300	65.74479	65.74479	65.74479	65.74479	65.74479	65.74479	65.74479	65.74479

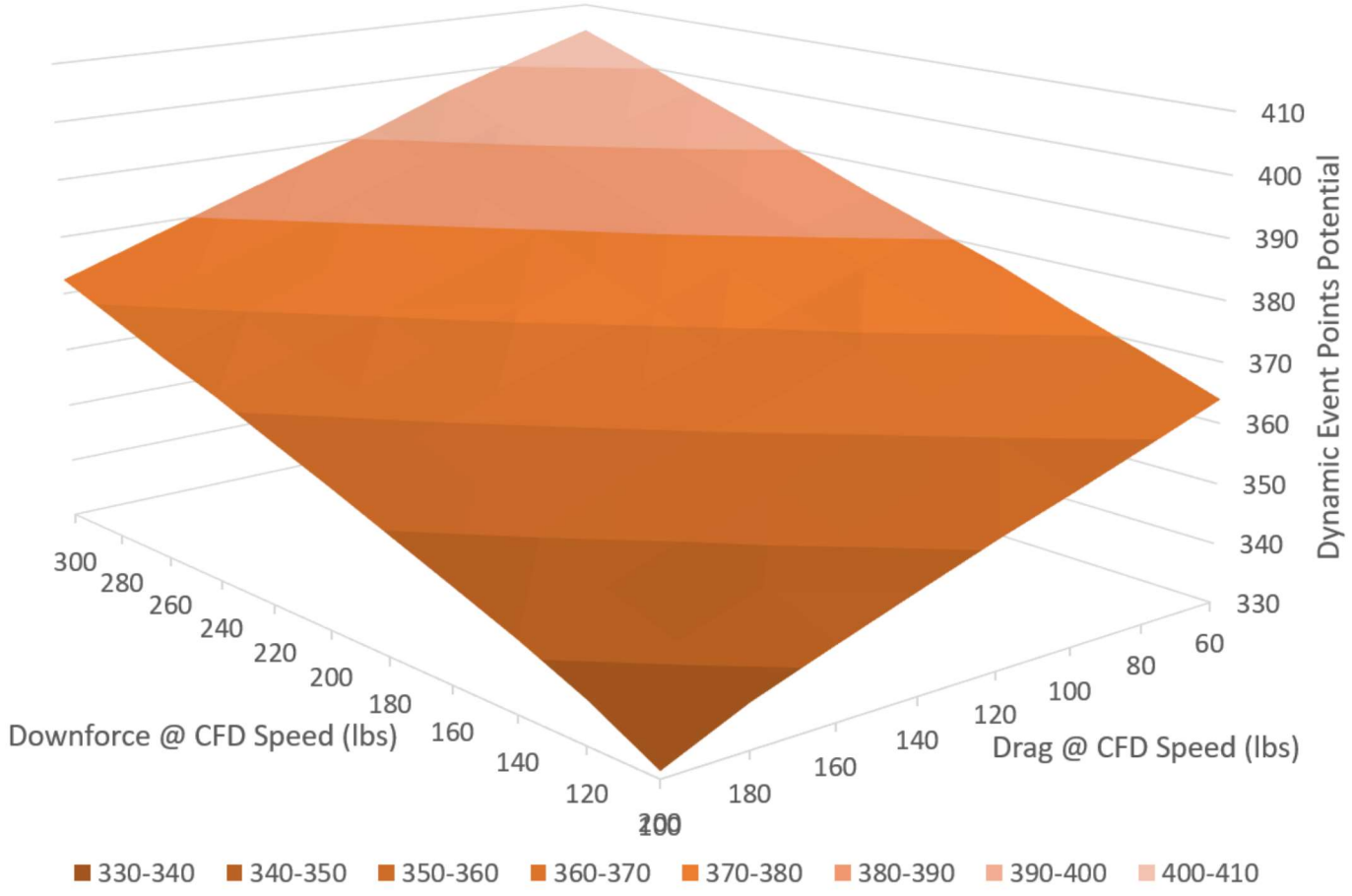
Accel Score		Drag @ 20 m/s [lbs]							
Absolute		60	80	100	120	140	160	180	200
Downforce @ 20 m/s [lbs]	100	84.45073	82.53759	80.6056	78.70772	76.7907	74.90571	73.00766	70.62013
	120	84.57202	82.70027	80.77626	78.86932	76.98817	75.20785	73.32347	71.37935
	140	84.85075	82.91203	81.02619	79.12523	77.25628	75.36642	73.52385	71.78384
	160	85.03896	83.13612	81.24017	79.3719	77.49793	75.62474	73.76459	71.96456
	180	85.27802	83.40426	81.37466	79.50426	77.71384	75.85696	74.00283	72.21206
	200	85.38714	83.5061	81.59446	79.71455	77.84925	75.97768	74.23856	72.42314
	220	85.60148	83.68325	81.83944	79.93548	78.07337	76.20677	74.35911	72.57102
	240	85.8121	83.89968	81.99698	80.13769	78.2598	76.41927	74.57455	72.79043
	260	85.98079	84.10097	82.20089	80.37311	78.47102	76.6204	74.78482	72.98297
	280	86.17543	84.18825	82.30371	80.41131	78.56953	76.71335	74.89538	73.17616
	300	86.24509	84.3387	82.48591	80.60542	78.78641	76.91347	75.07224	73.29416

Autocross Score		Drag @ 20 m/s [lbs]							
Absolute		60	80	100	120	140	160	180	200
Downforce @ 20 m/s [lbs]	100	80.54667	79.92734	79.29296	78.61612	77.92766	77.19857	76.41982	75.43194
	120	82.25065	81.68412	81.02252	80.35672	79.62583	78.87848	78.09253	77.24963
	140	83.911	83.18465	82.58141	81.8982	81.16804	80.40176	79.61148	78.79603
	160	85.73291	84.9579	84.17179	83.57065	82.83825	82.07959	81.26843	80.40893
	180	86.86304	86.31539	85.49186	84.89044	84.1309	83.35515	82.53861	81.6192
	200	88.41551	87.86487	87.02376	86.24198	85.47088	84.76877	83.86388	82.92952
	220	89.75389	89.20038	88.36214	87.51153	86.66709	85.93317	85.006	84.07558
	240	91.14277	90.45058	89.63976	88.66859	87.79699	86.95709	86.14419	85.19751
	260	92.30638	91.60696	90.78151	89.85362	88.94868	88.09752	87.27836	86.2953
	280	93.45749	92.75281	91.91818	90.97757	90.07144	89.16962	88.32948	87.35321
	300	94.62893	93.91185	93.06019	92.1203	91.18416	90.23728	89.26927	88.39254

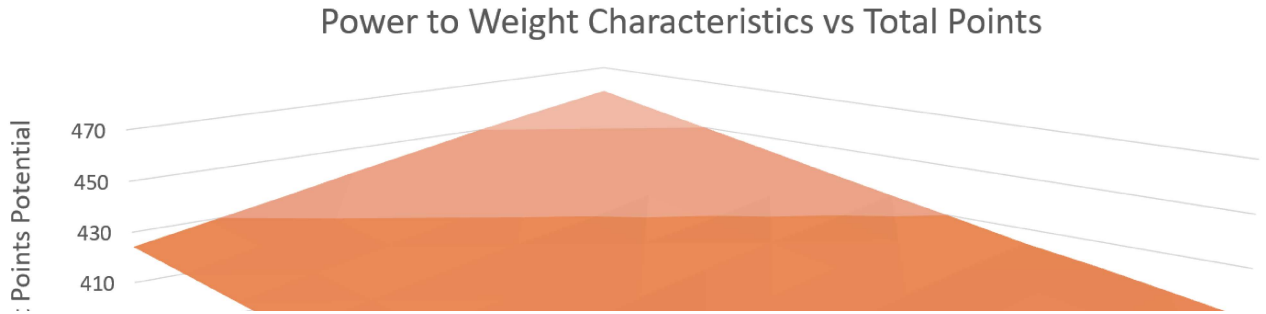
Endurance Score		Drag @ 20 m/s [lbs]							
Absolute		60	80	100	120	140	160	180	200
Downforce @ 20 m/s [lbs]	100	141.2031	139.3133	137.5488	136.0051	134.111	132.1287	130.0882	127.43
	120	143.0828	141.2632	139.4638	137.6997	135.814	134.0922	132.057	129.8676
	140	144.5701	143.0586	141.2206	139.3425	137.5528	135.6114	133.6163	131.7597
	160	146.4641	144.5867	143.105	141.1069	139.2936	137.3566	135.3302	133.2939
	180	148.1285	146.3977	144.5473	142.6251	141.0977	139.1397	137.1028	135.041
	200	149.4869	148.1353	146.3324	144.3172	142.5395	140.8451	138.8372	136.7488
	220	151.3133	149.9253	148.2279	146.1159	144.4236	142.2587	140.5472	138.4487
	240	153.0642	151.6422	149.8992	147.8365	145.9791	144.0525	141.9571	140.2347
	260	154.8661	153.3856	151.6695	149.5405	147.6056	145.7986	143.7225	141.5596
	280	156.6637	155.1732	153.4983	151.223	149.3448	147.4578	145.456	143.2873
	300	158.4472	156.947	155.3325	152.9176	151.077	149.1658	147.1249	145.0226

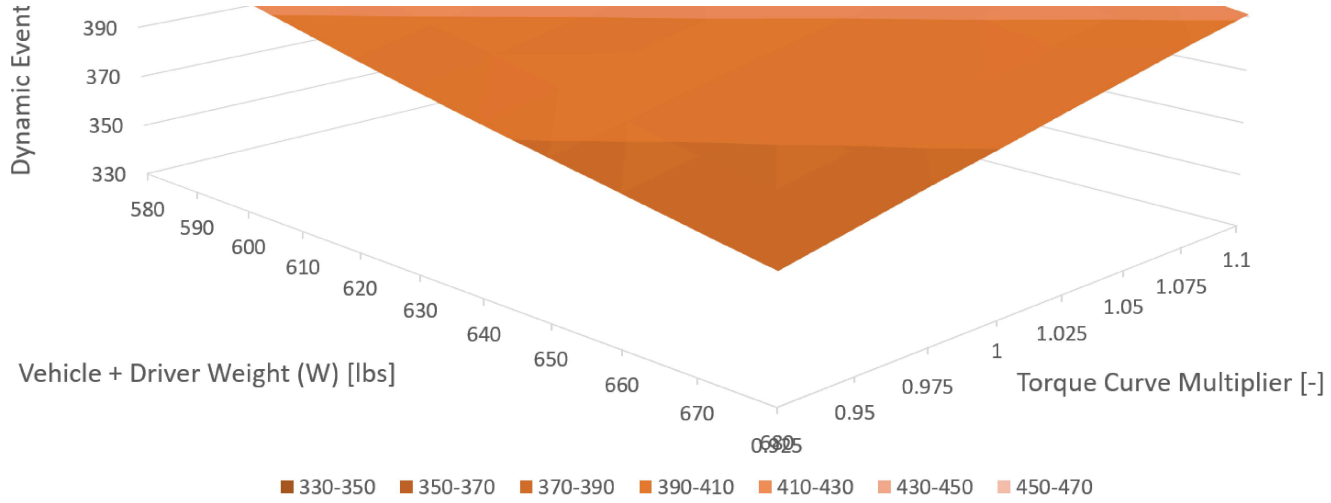
Some event-specific results displaying the sensitivity of vehicle performance to different aerodynamic lift and drag configurations

Aerodynamic Performance Metrics vs Total Points

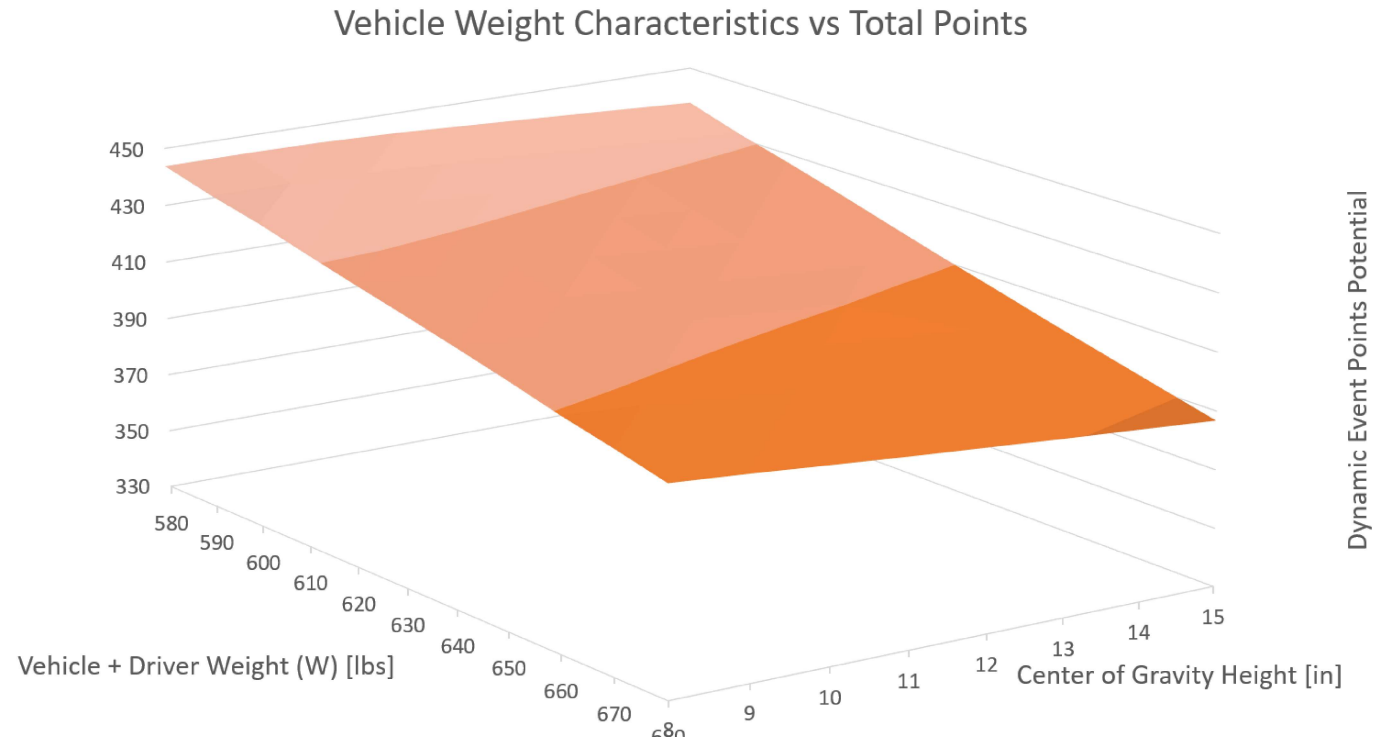


Full points summary of vehicle performance sensitivity to various aerodynamic lift and drag configurations





Full points summary of vehicle performance sensitivity to various power-to-weight ratio configurations



Full points summary of vehicle performance sensitivity to various weight and cg height configurations

Conclusion and Future Development

The general accuracy of the endurance lap when compared to logged data, as well as the reasonable results from the sensitivity analyses, demonstrate the usefulness of this lap simulation as a tool to evaluate and compare various design configurations. However, further validation can be carried out, including different track configurations and vehicle characteristics. For example, lap times could be taken with increasing amounts of ballast in the cockpit to evaluate whether the lap sim is properly predicting the proportional performance loss or not.

In addition, I would like to explore future opportunities to re-introduce transient effects into this simulation. Another simulation currently in the team's repertoire is a 7 DoF ride model, though it is still in need of some fine tuning. The velocity and acceleration outputs from the lap sim could be combined with a road surface profile as an input to this model, generating a time history of the normal loads on each tire throughout the course. This can be used to generate scaling factors for the GGV diagram as a function of position along the track. This updated GGV diagram can be used to re-evaluate the lap sim, with the results in turn being fed back into the ride model. This process can be repeated until the lap simulation results converge with the updated GGV predictions. This would be a "quasi-transient" method of sorts, and would ignore factors such as driver control models, but it can easily be incorporated into the existing simulation architecture and would provide a valuable approximation that can help evaluate the effects of dampers, center of pressure migration, and many more variables!

One of my friends was kind enough to take my files and documentation and create a GitHub repository! If you are interested in exploring the lap sim in greater detail, be sure to check that out!

View GitHub Repository

Back to Top

+44.7922.955425

Next it was time to narrow in on the desired range of available damping. For this car we focused on three primary modes of body motion: Pitch, heave, and roll. As a starting point, we built off of previous work done in 2019 with the help of Ohlins USA and their 4 post shaker rig.