Racing Line Optimization @ Race Optimal

Ricky Vesel — Race Optimal (http://www.raceoptimal.com/) — rickyv@raceoptimal.com

Race Optimal (www.raceoptimal.com) is a project aimed at creating an effective and fully automated process for racing line optimization in order to produce a database of racing line for racing circuits around the world. The ideal racing line is defined as the trajectory around a track that allows a given vehicle to traverse the circuit in the minimum time. In practice, it is an abstraction that varies with track, environmental conditions, vehicle type and condition, competitive traffic, and other factors. A certain extent of the talent possessed by the elite driver is the ability to perceive this optimal path and its variations, as well as to navigate it as quickly as possible. Until now, automated methods have struggled to match the performance of human-generated or human-guided racing lines [1]. This article describes the optimization engine and representation scheme used at Race Optimal to compute realistic and high performing racing lines for a wide variety of vehicle types.

Despite their abstract existence in the world of sport, there is significant use for concretely defined racing lines. Racing video games are quite popular, and require high quality racing lines in order to provide challenging AI competitors [1]. Racing lines have also a role in drivers' education, where they are used to develop visualization skills, as well as to provide a comparison of a driver's current approach with a potentially superior one. GPS data overlays onto track maps are currently used as driver aids and a logical extension of this approach is to display the generated optimal racing lines for comparison as well.

An optimization process that encompasses vehicle characteristics can further be used for race preparation for example to choose a downforce configuration or transmission setup, as well as to quickly familiarize a driver with a new circuit. When tackling the problem of racing line optimization, different simplifying assumptions have been applied. A 2D kinematic approach dictates that a vehicle's speed is governed by

$$v_{max} = \sqrt{\frac{r\mu(mg + F_{DF})}{m}} \tag{1}$$

where r is radius of curvature, μ is tire friction coefficient, m is vehicle mass, and F_{DF} is aerodynamic downforce. Maximizing the radius of curvature everywhere on the path will then maximize allowable vehicle speed everywhere. However, a shorter path, though having a smaller radius of curvature, may still take less time to traverse due to the shorter distance. Thus, one assumption applied in racing line optimization is that the optimal path will be a combination of the maximum curvature path (MCP) and the shortest distance path (SP). Producing the optimal linear combination of these paths is the accomplished by Braghin et al. [2]. This approach can be extended by dividing the track into subsections where the optimal tradeoffs between MCP and SP are optimized independently [3]. However, not all the approaches assume that the optimal racing line is a combination of the MCP and SP; in fact, other authors explored alternative methods of generating the racing line geometry, for example Euler spirals [4] or Bézier curves [5]. A simple point-by-point representation of the racing line may be also applied [3].

Once a racing line representation scheme is established, a fitness function is required to evaluate the candidate solutions. Fitness may simply be the measurement of path length or total curvature, but in order to capture the variation of the ideal racing line for different vehicles, some considerations about vehicle dynamics and power level are necessary. In [3], a robot driver is used in a high quality simulation to evaluate candidate racing lines. The present work also uses a computer simulation, but one that takes into account a much more limited model of vehicle dynamics in order to run as quickly as possible while still capturing the essence of several vehicle types.

A framework for racing line optimization should

1. Allow a high degree of geometric flexibility in racing line representation
2. Produce a racing line that appears realistic and smooth, without kinks or unnecessary undulations
3. Produce visually appealing and convincingly accurate racing lines for several vehicle types
4. Make full utilization of the available track surface, meeting all apexes and borders where necessary
5. Require minimal setup procedure allowing for the analysis of a large number of tracks and vehicles, with a fully automated solution process

By applying unique methods of geometric representation, optimization, and physics simulation, we believe these goals have been achieved. The methodology is described in the following sections.

Racing Line Representation

One of the biggest challenges was creating a racing line representation scheme that satisfies the above criteria, particularly (1), (2), and (4). Experimentation with different approaches revealed that all of those mentioned above fail to meet criteria (1) alone. To elaborate, a high degree of flexibility means that the racing line should be able to take on any conceivable shape that is physically realistic for a vehicle traversing a racetrack. Use of a predefined geometric shape, such as an Euler spiral, while interesting, is obviously too restrictive to satisfy this aim.

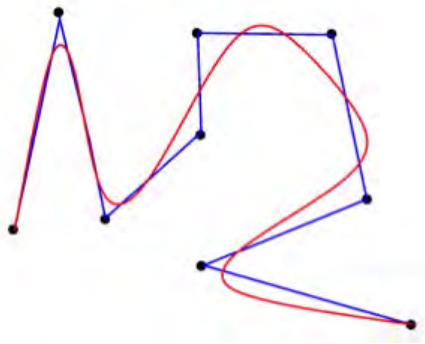

Fig. 1: Example of a Bézier curve, with control points (black) and the resulting curve (red) [5].

Fig. 2: Sample control point locations. The blue lines represent the possible locations of the attached control points (red). Adapted from [5].

This requirement should also obviously disallow a simple point-by-point construction of the racing line, with every point free to move independently; the unpredictable jumps from point to point would result in unphysical vehicle behavior and/or difficulty calculating the actual trajectory. It quickly becomes clear that a control point approach combined with some sort of smoothing function is necessary.

The authors in [5] utilize a Bézier curve (Figure 1), and this approach comes closest to satisfying our requirement. The control points, however, are restricted to the line segment perpendicular to the track on which they are placed, having a length of 120% of the track width, as shown in Figure 2. However, because Bézier curves typically do not intersect their control points, depending on the point density, there is no guarantee that this scheme will allow the racing line to reach the edges of the track, failing Criteria 5. Furthermore, limiting the control point positions to a line segment could result in an unacceptable likelihood that the fastest geometry might only be inadequately approximated by the limited set of potential Bézier curves available in this scheme, failing Criteria 1. Thus, in the current work, the control points are allowed to move more freely, within a circle of radius equal to four times the average track width, originating at initial user defined locations. An example is shown in Figure 3.

As part of fulfilling Criteria 3 to produce visually appealing racing lines, the line must smoothly connect to itself. This is important not only for the aesthetic quality, but because if that requirement is not enforced, the line could begin a circuit on a different path than with which it concludes, which is illogical in the context of lapping a racetrack. Initially the working solution was to forge a connection between the starting and ending points with an intermediate Bézier curve, which resulted in significant added complexity. Eventually, a more satisfactory solution was arrived upon, which utilizes a periodic smoothing spline [6].

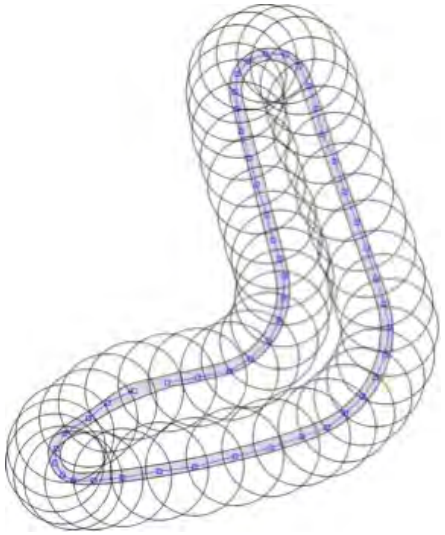

Fig. 3: Sample control point distribution (blue) and the circles representing their possible locations (black). Each control point is located at the center of its circle.

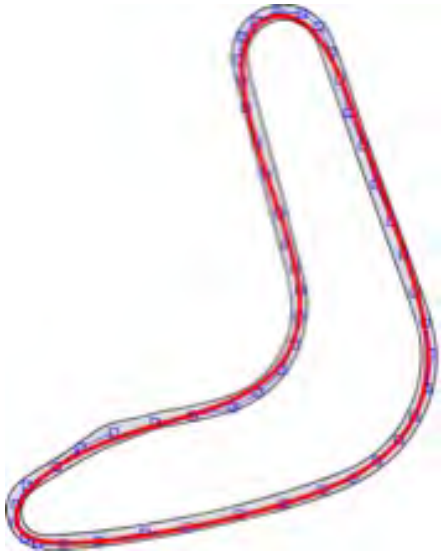

Fig. 4: Control points (blue) and the resulting periodic smoothing spline (red) using a smoothing factor of 0.25.

Periodic smoothing splines allow for a parameterized degree of smoothing, and automatically create a nicely connected curve which satisfies the following "periodic" end conditions over the interval $[x_1, x_n]$ [6]:

$$f(x_1) = f(x_n)$$

$$f'(x_1) = f'(x_n)$$

$$f''(x_1) = f''(x_n)$$

To obtain the racing line resulting from a set of control points, a parameter is introduced such that the x and y coordinate data are treated separately as a function of that parameter.

For example, control points $P_k = (x_k, y_k)$, $k \in \{1, \ldots, n\}$, are separated into $P(x,k) = (t_k, x_k)$ and $P(y,k) = (t_k, y_k)$, with $t_k \in \{1, 2, \ldots, n\}$. The coefficients of the periodic smoothing spline are then found for the x and y data separately, after which they are plotted together against t to form the smoothly connected racing line. The present work uses smoothing parameter $p = 0.25$. An example of the resulting (not optimized) racing line is shown in Figure 4.

Geometric Constraints

Because we use a control point scheme that allows for a highly variable racing line shape, some method must be employed to restrict the resulting line to the confines of the track. The first step in this process is to detect if the racing line intersects the start/finish line within the first 10% of its segments. Any line that fails this test is rejected automatically as invalid, thus ensuring that every candidate at least begins in an allowable position. The remaining task is to detect which points fall outside the bounds of the circuit, and apply an appropriate penalty. This was initially accomplished by counting the intersections between the borders and the racing line. Since it is known that the candidate begins inside the track, after one intersection it will be out of bounds, and will remain so until another intersection, and so on. Even after porting this functionality to C code, this process was unacceptably slow. A racing line typically has about 1000 points per mile, and track borders roughly the same point density. Evaluation often required tens of millions of intersection tests per solution.

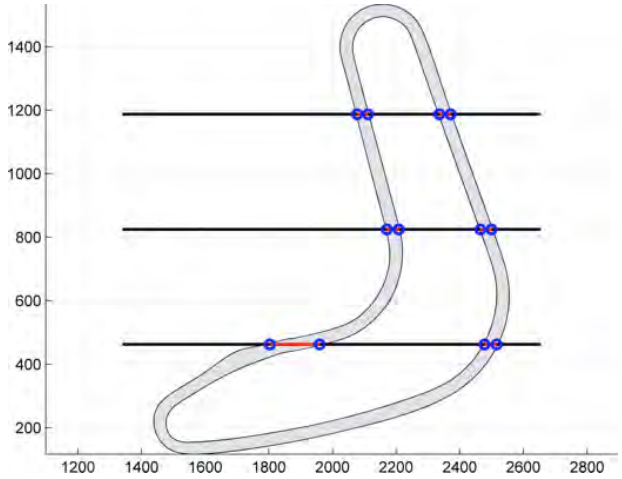

Fig. 5: Example of the horizontal slice method. Typically 10,000 or more horizontal lines are drawn at equal intervals (black) and the intersections with the borders are calculated (blue). The resulting valid x-intervals are shown in red. Three lines are shown here for demonstration.

A new method was devised to eliminate this bottleneck in the evaluation process, allowing for out-of-bounds points to be found very quickly, after a small initial cost at setup. In this method, several thousand horizontal lines are superimposed over the track borders, and the points of intersection with the borders are determined (Figure 5). A lookup table is produced that contains, for each line, the x-coordinates that contain the in-bounds sections of track. For a given point on the racing line, the table row with the closest y-coordinate can be quickly ascertained. That row can then be scanned to determine if the racing line point's x-coordinate is valid in any of the x-intervals for that row. Table 1 contains a sample of the lookup table corresponding to Figure 5.

Fitness Evaluation

Since this project requires the production of optimal racing lines for several vehicle types, vehicle simulation must obviously come into play to capture these differences. And because a flexible geometric scheme is utilized, there is no compelling reason to restrict oneself to SP or MCP solutions. This frees the process from as many assumptions about the optimal geometry as possible.

Tab. 1: Sample of in-bounds lookup table corresponding to the slices drawn in Figure 5.

y-coordinate	xleft,1	xright,1	xleft,2	xright,2
462.4	1802.6	1959.3	2477.1	2516.1
825.0	2170.7	2207.7	2465.3	2499.9
1187.6	2077.2	2111.1	2335.4	2370.9

Thus, vehicle simulation lap time is the core of the fitness function. The simulation process is outlined as follows:

1. Test for intersection with the start-finish line
2. Calculate the radius of curvature at every point, using

$$r = \left| \frac{(x'^2 + y'^2)^{3/2}}{x'y'' - y'x''} \right|$$

3. Calculate the maximum allowable velocity at every point, limiting to vehicle top speed
4. Beginning at the slowest point, iterate forward, limiting acceleration to that allowed by excess grip, vehicle power level, and aerodynamic drag
5. Iterate backward, limiting deceleration to that allowed by excess grip and aerodynamic drag
6. Enforce smooth input transitions
7. Calculate lap time
8. Count the out-of-bounds points and apply the appropriate penalty

Several items require further explication, beginning with (3). Maximum velocity is determined from the kinematic equation for centripetal force:

$$F_c = \frac{mv^2}{r} \tag{2}$$

This force is equal to the level of cornering force available,

$$F_c = \mu(mg + F_{DF}) \tag{3}$$

where F_{DF} is the aerodynamic downforce. From the well-known equation for aerodynamic lift, downforce can be modeled by $F_{DF} = 1/2\rho v^2 S C_L$. For a ground vehicle, however, the terms other than velocity can be grouped in a single parameter, k_{DF}, where $F_{DF} = k_{DF} v^2$. By setting Equations 2 equal to Equation 3 and plugging in this result, we can solve for velocity:

$$v = \sqrt{\frac{r\mu mg}{m - r\mu k_{DF}}}$$

Since the term in the denominator approaches zero as r_{max} approaches $m/(\mu k_{DF})$, this implies that for $r \geq r_{max}$, a vehicle with downforce can travel at an unlimited speed. Thus, when r is above r_{max}, the maximum velocity is taken to be the vehicle top speed.

Steps (4) and (5) translate the maximum allowable speed based on radius into realistic vehicle speed based on simplified vehicle dynamics. To limit the acceleration in the forward direction, the point at which speed is minimum is selected. The algorithm then looks to the velocity at the next point and computes the desired acceleration. The actual acceleration (and throttle input) is then restricted to that allowed by tire grip, engine power, and aerodynamic drag. After processing the entire path this way, the process is repeated in reverse, treating braking zones as acceleration zones in the backward direction. For braking, however, aerodynamic drag provides a contribution rather than a cost. Figure 7 demonstrates the output of steps 3-5 for the Mid-Ohio Sports Car Course, one of the circuits featured at RaceOptimal.com.

Initially this was the extent of the vehicle simulation. However, some Race Optimal users pointed out the unrealistically fast input transitions, as well as rapid input corrections mid-corner that would likely lead to instability. Since the vehicle simulations are published on Race Optimal in video form, these shortcomings needed to be addressed. A smoothing process was developed that limits the rate of input transition depending on the cornering load, where maximum cornering load requires the slowest rate of throttle or brake transition. The details of the smoothing algorithm are beyond the scope of this article, but an example of the results are shown in Figure 7 and Figure 8. Including this limit in the optimization also improved the shape of the final racing line, since the simulation no longer had the luxury of traversing a line in a way that requires instantaneous input corrections.

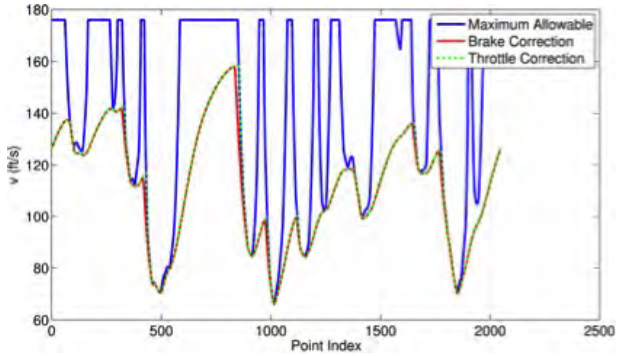

Fig. 6: Progression of simulated vehicle velocity, starting with the maximum allowable and showing corrections for acceleration and braking zones. Simulation is for a Mazda Miata at the Mid-Ohio Sports Car Course.

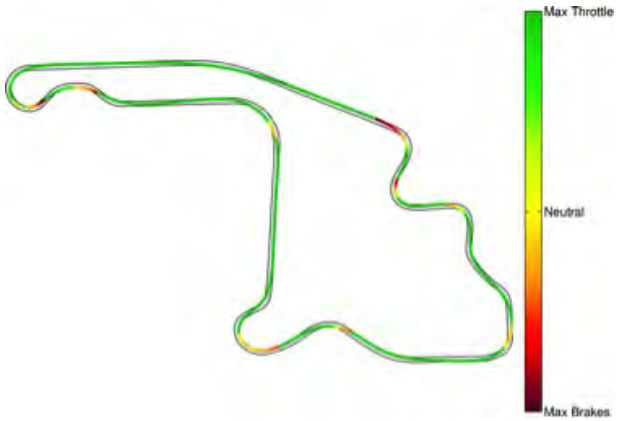

Fig. 7: Throttle and brake for Mazda Miata at Mid-Ohio Sports Car Course without smoothing correction.

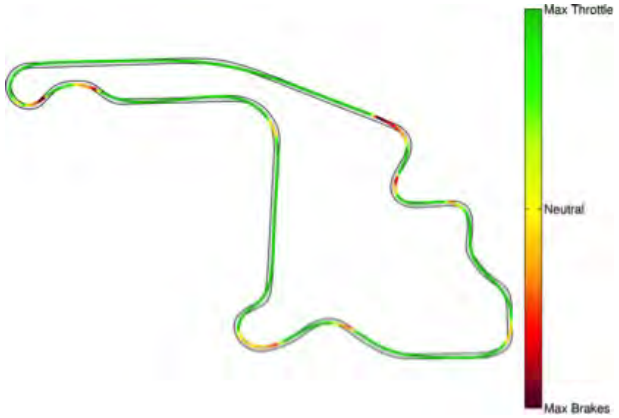

Fig. 8: Throttle and brake for Mazda Miata at Mid-Ohio Sports Car Course with smoothing correction.

Genetic Algorithm

We experimented with several approaches, including particle-swarm optimization and genetic algorithms with various parameters. The current approach is a standard genetic algorithm with a supplemental subprocesses called “healing” used to improve performance. A binary-encoded genome is utilized encoding 75 control points. Each control point has an offset from its origin defined in radial coordinates, with 14 bits encoding r and Θ. Selection from the parent population for breeding is made using roulette wheel, where the likelihood of selection is proportional to candidate fitness. Bit crossover probability between mating genomes is 75%, and mutation probability varies with generation, but is at most 0.2%. Each generation, a child solution set is created equal in size to the parent generation. A simple replace-worst strategy is utilized, where the worst 90% of parent solutions are replaced by the best 90% of child solutions.

The maximum number of generations is set to 15000, but the optimization can end as early as 10000 if no substantial lap time improvement is being obtained. In order to combat premature convergence, after 2000 generations the population is re-randomized using the current best solution as the seed.

This approach provides generally good results, but not good enough to satisfy the Criteria 3, which requires visually appealing and convincingly accurate racing lines. The optimization would frequently miss apexes by a small to medium distance, and would do this consistently at certain turns on some tracks, despite the fact that clipping the apex resulted in a quicker lap. Although the difference in lap time was rarely more than one or two tenths of a second, such errors are noticeable in the resulting racing lines, and undermine the credibility of the results. It was eventually determined that this problem was caused by the solutions becoming overly sensitive to out-of-bounds points at the entrances and exits of some turns. The racing line converged on the border at the entrance or exit before the apex, and by that time moving closer to the apex pushed the other sectors out of bounds. Since out-of-bounds points are penalized stiffly in order to prevent them from appearing in the final solution, the result was that the optimization was unable to meet the racing line with the apex of some turns.

In order to address this shortcoming, a "healing" sub-process was developed that is applied to a certain percentage of offspring solutions. To heal a racing line, control points closest to concentrations of out-of-bounds points are incrementally shifted toward the in-bounds direction in an attempt to bring the out-of-bound points out of bound back onto the track. This process involves recalculating the racing line many times for a given solution, and therefore increases the computational burden substantially. It was found that limiting the likelihood of healing was beneficial both to processing time and final solution quality, as healing every solution tended to interfere too much with the genetic algorithm itself. Although the optimization still occasionally produces a missed apex, the frequency of this occurring is greatly reduced with the incorporation of the healing sub-process.

Results

As an example, we applied our optimization scheme to the Mid-Ohio Sports Car Course. In particular, in addition to the normal simulation-based optimization, we also performed optimizations seeking the SP and MCP solutions—the latter maximizing the sum of curvature as the fitness function. Figure 9 shows the MCP and SP solutions compared with the simulation-based optimization in a section of track called the Keyhole. The black-filled area represents the possible linear combinations of the MCP and SP solutions. Figure 10 shows the same results, but for the section of track made up of Turns 6-10, a series of connected esses. It can be seen that over most of this section the MCP and SP solutions are very similar. The simulation-based racing line enters Turn 6 within the MCP-SP area, but thereafter follows a path that deviates substantially from what would be allowed by any linear combination of the two.

Lap times for the three optimizations are shown in Table 2. The simulation-based result achieved a lap time of 1'43.42. The National Auto Sport Association (NASA) currently lists the track record (eligible laps occur during a timed points race) for Spec Miata vehicles at Mid-Ohio as 1'45.655[1]. This is over two seconds slower than the optimal lap time achieved in the simulation; investigations are underway to determine if this discrepancy is a result of optimistic vehicle parameters or possibly other inaccuracies in the simulation. Nevertheless, the simulation-based result beat the MCP solution by over 4 seconds, and the SP line by over 5 seconds. However, no optimization of the tradeoff between MCP and SP paths was performed. The time elapsed through Turns 6-10 can be measured for a better comparison, since there is little variation between the MCP and SP lines, and the MCP line alone is a plausible route through that sequence. The sector times are shown in Table 2, where the simulation-based solution is 0.67 seconds quicker than the MCP.

Bearing in mind the fourth criteria, to capture the differences in racing line for different vehicles, the same section from Mid-Ohio is shown in Figure 11, via a screen capture from raceoptimal.com. The four vehicle types are the Mazda Miata, Porsche 911 GT3 RS 4.0, a generic motorcycle-type vehicle, and a theoretical F1 car.

[1] http://www.nasamidwest.com

The optimization clearly produces different paths for the four vehicles. The entire track must also be considered in order to assess the accomplishment of the criteria listed in the introduction, and this is shown in Figure 12 for the Mazda Miata. The result is a fluid racing line that touches the borders in the expected places, and takes a line very similar to what one can observe in professional onboard footage. Thus, the initial five criteria for an automated process of realistic racing line optimization have been satisfied.

Conclusions

A racing line optimization procedure has been developed that produces plausible real-world results that are visually pleasing and can capture the behavior of different vehicle types. The procedure does not rely on assumptions about the geometry of the ideal racing line. It is hoped that future authors will free themselves from the restrictions of predetermined geometry or racing line characterization when undertaking the challenge of racing line optimization.

As of writing, 62 circuits with 145 individual configurations have been analyzed on RaceOptimal.com, with unique racing lines and video simulations for each of the four vehicles. Thus, the usefulness of the present method as a means of developing a racing line database has been established.

Future Work

The current track geometry is modeled in two dimensions, meaning that elevation changes and track camber are not considered, accordingly, future work will include 3D considerations. Also, a more sophisticated vehicle model can be used in order to more accurately capture the interaction between the racing line and vehicle dynamics, as well as to include variable vehicle parameters, such as gearing, downforce configuration, and tire hardness in the optimization process.

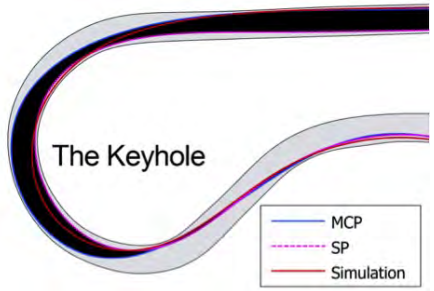

Fig. 9: The Keyhole section of Mid-Ohio Sports Car Course, showing the MCP, SP, and simulation based optimization results. The black region represents the track accessible by a linear combination of the MCP and SP solutions. The red line is the optimized path for a Mazda Miata. The racing line is the vehicle centerline, so a gap exists between the racing line and the track borders even at the nearest point.

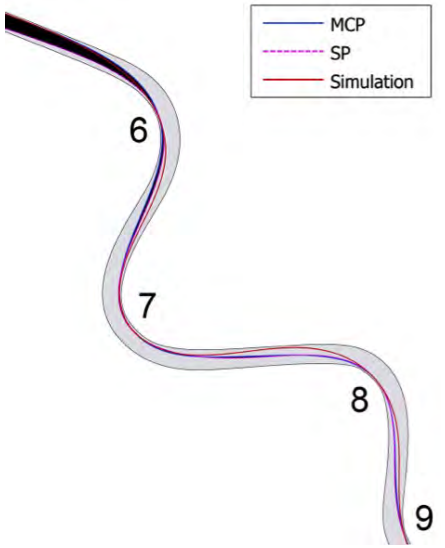

Fig. 10: Series of esses beginning with Madness from the Mid-Ohio Sports Car Course, showing the MCP, SP, and simulation based results. Clearly, the simulation based path requires track areas well outside the bounds of the MCP and SP solutions.

Tab. 2: Lap and sector times for the three optimizations.

Sector	MCP Time	SP Time	Simulation-Based Time
Full Track	1’47.67	1’48.86	1’43.42
“Madness” Section	19.21	19.57	18.54

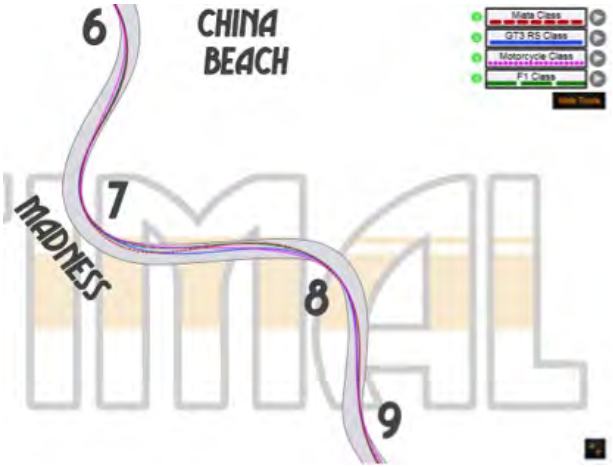

Fig. 11: Sample of the final racing line map for Turns 6-9 at Mid-Ohio Sports Car Course —- Club Circuit. The individual lines can be toggled on and off.

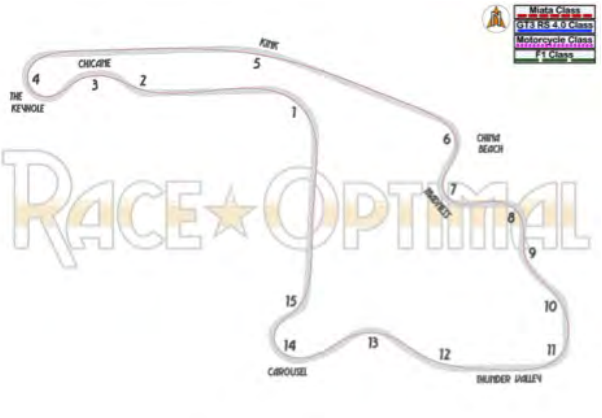

Fig. 12: Sample of the final racing line map for the Mazda Miata at Mid-Ohio.

References

[1] Stefano Lecchi. *Artificial intelligence in racing games*. In CIG’09: Proceedings of the 5th international conference on Computational Intelligence and Games, pages 1–1, Piscataway, NJ, USA, 2009. IEEE Press.

[2] F. Braghin, F. Cheli, S. Melzi, and E. Sabbioni. *Race driver model*. Comput. Struct., 86(13-14):1503–1516, 2008.

[3] L. Cardamone, D. Loiacono, P.L. Lanzi, and A.P. Bardelli. *Searching for the optimal racing line using genetic algorithms*. In Computational Intelligence and Games (CIG), 2010 IEEE Symposium on, pages 388–394, aug. 2010.

[4] Y. Xiong. *Racing Line Optimization*. Thesis. Massachusetts Institute of Technology, 2010.

[5] M. Botta, V. Gautieri, D. Loiacono, and P.L. Lanzi. *Evolving the Optimal Racing Line in a High-End Racing Game*. In Computational Intelligence and Games (CIG), 2012 IEEE Symposium on, pages 108–115, aug. 2012.

[6] N. Y. Graham. *Smoothing With Periodic Cubic Splines*. The Bell System Technical Journal 62.1 (1983): 101-10.

About the author

Ricky Vesel received his MS in Aerospace Engineering from The Ohio State University, where he specialized in wind turbine optimization. He became interested in applying similar techniques in searching for the ideal racing line, and solidified the concept over the course of a six week solo motorcycle trip. Race Optimal was created as a result. He currently lives in Vietnam and is an avid table tennis player.

Homepage: http://www.raceoptimal.com/
Email: rickyv@raceoptimal.com