EasyChair Preprint

№ 11507

Double Gradient Method: a New Optimization Method for the Trajectory Optimization Problem.

Alam Rosato Macêdo, Ebrahim Samer El Youssef and Marcus V. Americano da Costa

December 10, 2023

Double Gradient Method:

A new optimization method for the trajectory optimization problem.

Alam Rosato Macêdo[1], Ebrahim Samer El youssef[2] and Marcus V. Americano da Costa[1]

[1] Universidade Federal da Bahia, Bahia, Brazil
{alam.rosato, marcus.americano}@ufba.br
[2] Universidade Federal de Santa Catarina, Santa Catarina, Brazil
ebrahim.el.youssef@ufsc.br

Abstract. In this paper, a new optimization method for the trajectory optimization problem is presented. This new method allows to predict racing lines described by cubic splines (problems solved in most cases by stochastic methods) in times like deterministic methods. The proposed Double Gradient Method (DGM) is not affected by the dimensionality of the problem. Comparison of the results with data collected from professional drivers has shown that the DGM is reliable for lap time simulations with race line optimization. It can help drivers find the fastest racing line, be used for embedded algorithm development or for autonomous vehicle competitions.

Keywords: Lap Time Simulation, Optimization, Autonomous Racing Vehicle.

1 Introduction

Over the years, simulators have been developed to help drivers find the fastest racing line [1] or to help develop embedded algorithms, such as stability control [2]. Lap time simulators with race line optimization using nonlinear programming have become attractive since the advances in computing capacity, as they can perform optimizations of hours [3, 4] in a few minutes [5, 2, 6, 7, 8].

In 2014, F. Biral and R. Lot [5] developed a method for predicting the optimal path using the three-dimensional coordinates of the track as an implicit reference in the equations of the vehicle model. Using Darboux's trihedron, they modeled the circuit in curvilinear coordinates, which in turn were used to derive the vehicle's motion equations. The proposed method can optimize racing lines in less than a minute but requires the equations to be formulated at a symbolic level to derive the model which is inserted into the solver.

The formulation of equations at the symbolic level requires information that is not always available to engineers from, for example, the Brazilian automotive categories, especially the semi-professional ones. Data for tire modeling are particularly difficult to obtain, even in professional categories. Stochastic methods [9, 10] and derivative-

free methods [11] can solve optimization problems without having to fully model the tires of the vehicle [1, 12, 13], but are not as fast as deterministic methods [9].

Autonomous vehicle competitions are an interesting platform to develop control algorithms to work at the limit of handling [14] aiming at more reliable embedded systems for road vehicles. Trajectory prediction time is a key factor for such competitions [15, 16]. Simplifying the vehicle model and precomputing the racing line for a new track using cubic splines [13] is one strategy to reduce the trajectory prediction time. As in regular semi-professional competitions where tire data is not available, trajectory optimization for simplified models with cubic splines path description falls back to stochastic or derivative-free methods.

Based on foregoing, this paper proposes an optimization method based on the gradient descent concept and inspired by Neural Networks to solve trajectory optimization problems: Double Gradient Method (DGM). This new method allows us to predict racing lines described by cubic splines in times similar to deterministic methods. The results of the developed method were compared with the results obtained by the following methods: Genetic Algorithm (GA) [17, 18], Differential Evolution (DE) [19, 20], Nelder-Mead [21, 22, 23] and Trust Region [24, 25, 26, 27]. These methods were chosen because they can solve the optimization problem without using the derivative of the objective function, which is not available in cubic spline path optimization problems.

The main contributions of this work are:

1. Development of a new method to solve trajectory optimization problems graphically i.e., manipulating a cubic spline that describes the vehicle path.
2. Application of this method on a race line prediction problem.
3. Comparison of the DGM with four other methods capable of solving race lines graphically.
4. Results validation comparing with professional driver performance.

This paper is organized as follows: Section 2 presents the optimization problem for a path interpolated by a cubic spline. Section 3 develops the proposed Double Gradient Method, and simulation results are presented in Section 4. Section 5 contains the conclusions, applications of the algorithm, limitations of the algorithm, and future works.

2 Optimization problem

The DGM presented in this paper is applied to the following problem: Given a vehicle model with one degree of freedom and a trajectory described by cubic splines, determine the path that gives the shortest lap time on a circuit by directly changing the control points of the cubic splines (see **Fig. 1**).

The optimization problem in the standard formulation is given by:

$$\min_{u} Fob(u) = \sum \frac{2 \cdot s_i(u)}{\dot{x}_i(u) + \dot{x}_{i-1}(u)} \tag{1}$$

Subject to:

$$u_i \leq u \leq u_s \tag{2}$$

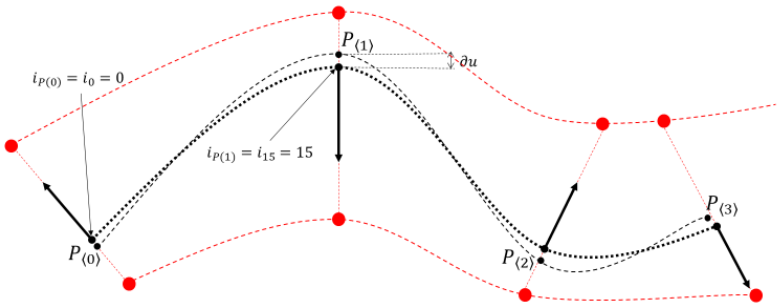

Fig. 1. Updating the trajectory control points position during the optimization process. The control coordinates (in red) are interpolated by the t parameters of the u vector of decision variables to generate the cubic spline control points $P\langle n \rangle$. p_i are the interpolated points between control points.

The objective function $Fob(u)$ is the numerical integration of the path distance s_i as a function of velocity (lap time), where s_i is the linear distance between two consecutive discrete points $\overline{p_{i-1}\, p_i}$ describing the vehicle path. The distance s_i cannot be integrated as a function of the spline by Bézier curves [28], since it is an elliptic integral, hence the use of a linear approximation between the interpolated points, which requires an appropriate interpolation mesh (less than 1m between two consecutive points, for errors up to the third decimal place). $\dot{x}_i$ represents the speed and is determined according to the model used to represent the vehicle on the track.

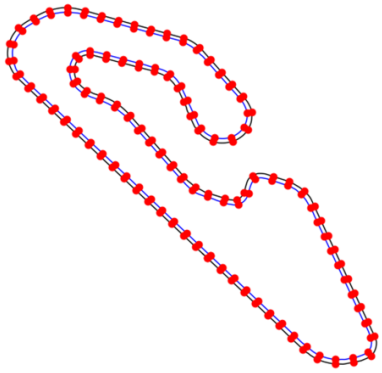

Fig. 2. In red, the control coordinates for generating the path for the Goiânia circuit.

The interpolation of the trajectory is based on control points that are uniformly distributed along the track. The control points must be homogeneously distributed due to the interpolation properties of the spline by Bézier curves. The interpolated points are distributed as a function of the linear interpolation of the Bézier points [29], so an irregular distribution of the control points leads to an irregular distribution of the in-

terpolated points. Since the interpolated points correspond to the problem discretization in space domain, it is necessary that they are distributed as homogeneously as possible to ensure an adequate mesh.

The u vector of decision variables contains the linear interpolation parameters of the pairs of control coordinates shown in **Fig. 1** and **Fig. 2**, which form the control points for spline interpolation, so problem (1) has $j-1$ degrees of freedom for a u vector with j parameters t. The bounds for the u vector are: $u_i = 0$ and $u_s = 1$.

The $Fob(u)$ solution requires the vehicle model simulation on the track, so the optimizer cannot use the objective function itself to solve the problem using deterministic methods. The $Fob(u)$ solution is not available for evaluation until the simulation is completed, so probabilistic methods and derivative-free methods are better suited to solve it, despite the high time cost. The "Double Gradient" method proposed in this paper was inspired by Neural Network topology and aims to take advantage of the use of gradients in solving optimization problems: Speed.

The requirements for the method are:

- The simulation time of the developed algorithm must be comparable to the simulation times of deterministic algorithms – a few minutes [5, 2, 6, 7, 8] – to allow its application during the shakedown of racing vehicles or during competitive events.
- The algorithm must optimize the trajectory without using the vehicle's dynamic equations, which allows for greater flexibility in terms of the simulation method used to represent the vehicle on the track. This flexibility allows the use of complex models with as much vehicle information as possible [6] or simplified models that require minimal vehicle data [13].
- The algorithm must be able to solve the problem graphically by directly modifying the cubic spline that defines the path. This requirement is a direct consequence of the previous requirement.

3 Double Gradient Method

Artificial Neural Networks (ANN) are the electrical equivalent of the biological neural networks from which they were inspired [17]. Neurons are represented by linear functions, while synapses are represented by nonlinear inhibitory functions that limit the signal amplitude processed in a cell.

The method developed in this section was inspired by a particular topology of Artificial Neural Networks: the single-layer network with error back-propagation, as shown in **Fig. 3**.

In matrix form, the topology of **Fig. 3** is represented as follows:

$$Y'(k) = \varphi(W(k) \cdot U(k)) \tag{3}$$

Where $Y(k)$ and $Y'(k)$ are the reference and the output of the network in k iterations. W and U are the weights matrices and the inputs matrices respectively and φ is the activation function. The adaptation law is given by the gradient rule:

$$e = Y - Y' \tag{4}$$

$$\frac{\partial e}{\partial W} = -\frac{\partial Y^{'}}{\partial W} = -\frac{\partial \varphi(W \cdot U)}{\partial W} \tag{5}$$

$$W(k+1) = W(k) - \alpha \cdot \frac{\partial e}{\partial W} \tag{6}$$

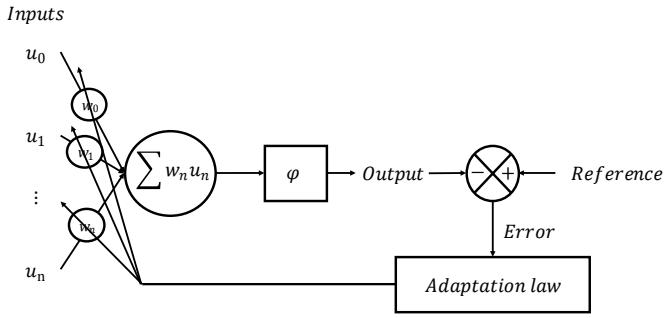

Fig. 3. Topology of an artificial neural network with error backpropagation

In order for the neural network's W weights to adapt, the network is presented with a set of U input data and their respective reference outputs Y during training. Only one input is available for solving the optimization problem and there is no reference to the error, yet the Double Gradient Method uses the basic topology presented for constructing the algorithm.

For application in solving the problem of trajectory optimization by control points for spline interpolation, the topology of **Fig. 3** was modified according to **Fig. 4**. The following equations (7 to 15) were developed to emulate the specialization phenomenon of a Neural Network by adjusting the weights to determine the best path. The resulting topology is not an ANN (there is no learning process), but an analog for optimization problems.

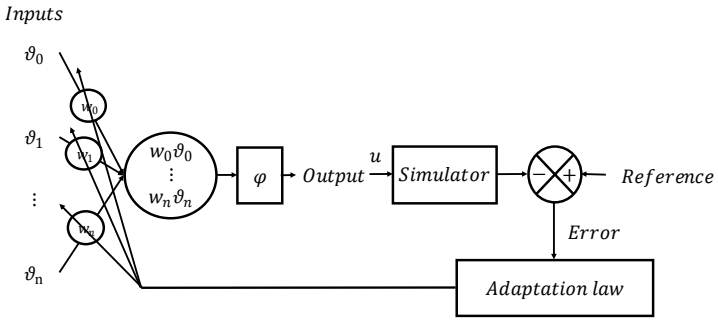

Fig. 4. Basic topology of the Double Gradient Method

In **Fig. 4**, the vector ϑ of inputs (u in the neural network) is an invariant fixed reference (e.g. the track center line) during the optimization process and the vector u be-

comes both the "network" output and the input for the simulator running the vehicle model, i.e., it becomes the vector of decision variables of the optimization problem.

In matrix form:

$$u\langle n\rangle(k) = \varphi(W\langle n\rangle(k) * \vartheta\langle n\rangle(k)) \tag{7}$$

Where $*$ denotes the row-wise product of the W and ϑ matrices such that u is an $n \times 1$ matrix. Three gradients are considered for the construction of the adaptation law:

$$\partial P_{ft}\langle n\rangle(k)_{[n\times 1]} = P_{ft}\langle n\rangle(k)_{[n\times 1]} - P_{ft}\langle n\rangle(k-1)_{[n\times 1]} \tag{8}$$

$$\partial u\langle n\rangle(k)_{[n\times 1]} = u\langle n\rangle(k)_{[n\times 1]} - u\langle n\rangle(k-1)_{[n\times 1]} \tag{9}$$

$$\partial P_{\ddot{y}}\langle n\rangle(k)_{[n\times 1]} = P_{\ddot{y}}\langle n\rangle(k)_{[n\times 1]} - P_{\ddot{y}}\langle n\rangle(k-1)_{[n\times 1]} \tag{10}$$

Where P_{ft}, defined by equation (11), is a function that accounts for the time gain/loss between the control points of the spline that forms the trajectory and $P_{\ddot{y}}$, defined by equation (12), is a function that accounts for the variation in lateral acceleration of the vehicle between the spline control points that form the trajectory. Both equations were established empirically. Equation (12) was established to emulate the driver's pace on the track and the second and third terms of equation (11) provide more factual trajectory results when defined in this manner.

$$P_{ft} = \begin{cases} Fob(u)^2 \cdot \sum_{i_{P(n-1)}}^{i_{P(n)}} \frac{2 \cdot s_i}{\dot{x}_i + \dot{x}_{i-1}} \cdot \sum_{i_{P(n)}}^{i_{P(n+1)}} \frac{2 \cdot s_i}{\dot{x}_i + \dot{x}_{i-1}}, & if\ n = 1, \cdots, j-2 \\ Fob(u)^2 \cdot \sum_{i_{P(n-1)}}^{i_{P(n)}} \frac{2 \cdot s_i}{\dot{x}_i + \dot{x}_{i-1}} \cdot \sum_{i_{P(0)}}^{i_{P(1)}} \frac{2 \cdot s_i}{\dot{x}_i + \dot{x}_{i-1}}, & if\ n = j-1 \end{cases} \tag{11}$$

$$P_{\ddot{y}} = \begin{cases} \sum_{i_{P(n-1)}}^{i_{P(n)}} |\ddot{y}_i - \ddot{y}_{i-1}| + \sum_{i_{P(n)}}^{i_{P(n+1)}} |\ddot{y}_i - \ddot{y}_{i-1}|, & if\ n = 1, \cdots, j-2 \\ \sum_{i_{P(n-1)}}^{i_{P(n)}} |\ddot{y}_i - \ddot{y}_{i-1}| + \sum_{i_{P(0)}}^{i_{P(1)}} |\ddot{y}_i - \ddot{y}_{i-1}|, & if\ n = j-1 \end{cases} \tag{12}$$

Thus, the algorithm adaptation law is given by:

$$W(k+1) = \begin{cases} W(k) - \alpha(k) * sgn\left(\partial P_{ft}(k)\right) * sgn\left(\partial u(k)\right), & if\ \kappa_{P\langle n\rangle} \geq 0.002 \\ W(k) - \alpha(k) * sgn\left(\partial P_{\ddot{y}}(k)\right) * sgn\left(\partial u(k)\right), & if\ \kappa_{P\langle n\rangle} < 0.002 \end{cases} \tag{13}$$

The updating of the weights of the algorithm, as shown in equation (13), depends only on the sign of the gradient. The optimization of the trajectory is done by changing the location of the spline control points that form the path (see **Fig. 1**) .The value of the gradient is not important and makes it difficult to control the degree of change

of the trajectory between iterations of the optimization process, so only the sign of the gradient is needed for the adaptation law.

Comparing equations (13) and (6), we find that the weight updating step in (6) decreases as the reference error decreases, while this step in equation (13) is constant since there is no error parameter. To mimic the weighting update process, some features of equation (13) need to be analyzed:

- The weights are updated based on $\partial P_{ft}\langle n\rangle$ if the path curvature at the point $P\langle n\rangle$ is such that it can be considered a curve ($\kappa \geq 0.002$). Under this condition, small changes in the position of the control points can produce reasonable gains/losses in lap time.
- The weights are updated based on $\partial P_{\ddot{y}}\langle n\rangle$ if the path curvature at the point $P\langle n\rangle$ is such that it can be considered a straight line ($\kappa < 0.002$). In this condition, small changes in the position of the control points do not effectively affect the lap time. In this case, better results are obtained by reducing the variation of the lateral acceleration $\ddot{y}$.
- Gradients serve only as a "compass" to indicate the direction of the update weights. The update step is defined by the update rate α.
- The second term of equation (13) is always different from zero at all points except the trivial solution points. As the function approaches the suboptimal point, it stabilizes and oscillates indefinitely around that point.

Since we know that α is the only parameter that changes the update step of the weights, and that equation (1) stops progressing when approaching a suboptimum, the update rate α is treated as follows:

$$\alpha(k+1) = \begin{cases} \dfrac{\alpha(k)}{2}, & if\ \varsigma > 0\ and\ \varsigma\ \%\ nk = 0\ and\ \alpha > 0.0001 \\ \alpha(k), & elsewhere \end{cases} \tag{14}$$

Where the parameter ς indicates how many consecutive k iterations $Fob(u)$ does not progress and % represents the remainder of the division ς/nk. The parameter nk specifies how many iterations are allowed without result progress. The limit of 0.0001 is fixed because path changes at the level of tenths of a millimeter are meaningless for the lap time.

Finally, the φ function ensures that $0 \leq u\langle n\rangle < 1$ by the modified sigmoid according to equation (15), where $f = (W * \vartheta)$:

$$\varphi(f) = \frac{1}{1 + e^{-15.637 \cdot f + 7.819}} \tag{15}$$

4 Results

In this work, data collected from data loggers in competition vehicles were used to validate the simulated data (see **Fig. 5** to **Fig. 8**). Track data obtained from two professional drivers at the Autódromo Internacional Ayrton Senna (Goiânia) were used for validation. Vehicles and competition cannot be disclosed for confidentiality reasons.

Four already available methods were compared with the Double Gradient Method: Genetic Algorithm, Differential Evolution, Nelder-Mead and Trust Region. These methods were chosen because they can optimize problems without requiring the derivative of the objective function. This is the main feature for solving optimization problems with racing lines by cubic splines.

The Goiânia circuit was modeled by 93 pairs of control coordinates (see **Fig. 2**). **Table 1** shows the results obtained after the simulations with the four selected optimization methods and the Double Gradient Method developed and presented in this work. The proposed method was the only one that could solve the problem within a time window of minutes, comparable to that of the deterministic methods, which range from about 1 to 10 minutes [8]. To valid data analysis, all data were compared taking the inner edge of the track as a fixed reference. The reference vector ϑ was used as the initial guess for the four selected methods. Only the Double Gradient data are presented so as not to overload the figures.

Table 1. Simulations results compared to the driver's performance.

Method	Simulation time	Lap time [s]	Difference [s]
Differential Evolution	06:37:50	93.29	0.37
Genetic Algorithm	19:55:22	96.72	3.80
Double Gradient	00:07:29	93.30	0.38
Nelder-Mead	01:15:29	94.23	1.31
Thrust Region	01:59:45	94.30	1.38
Driver 1	NA	92.92	0.00
Driver 2	NA	93.20	0.28

The following figures show the metrics comparisons between the proposed method and professional driver's performances:

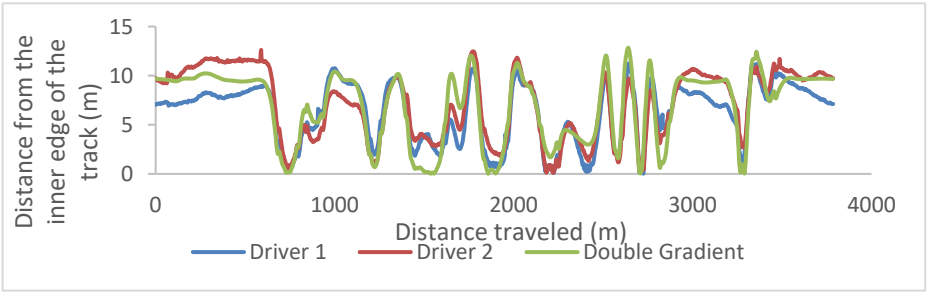

Fig. 5. Vehicle position along the track in relation to the track inside edge.

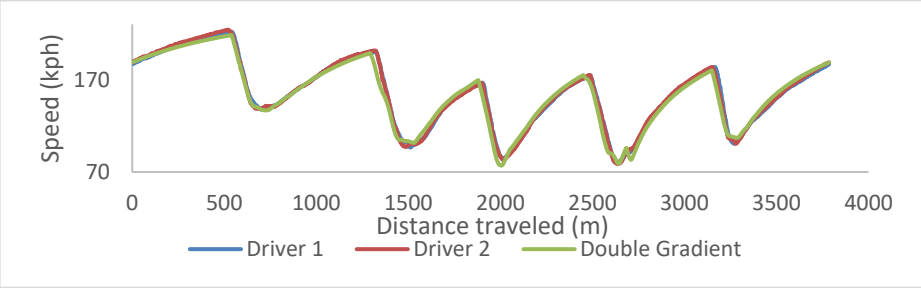

Fig. 6. Vehicle speed along the track.

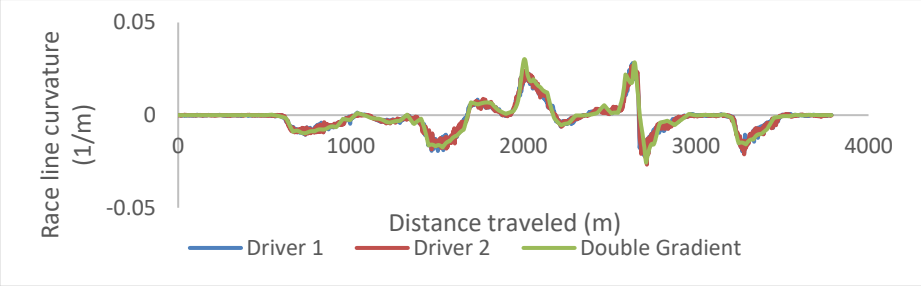

Fig. 7. Race line (path) curvature

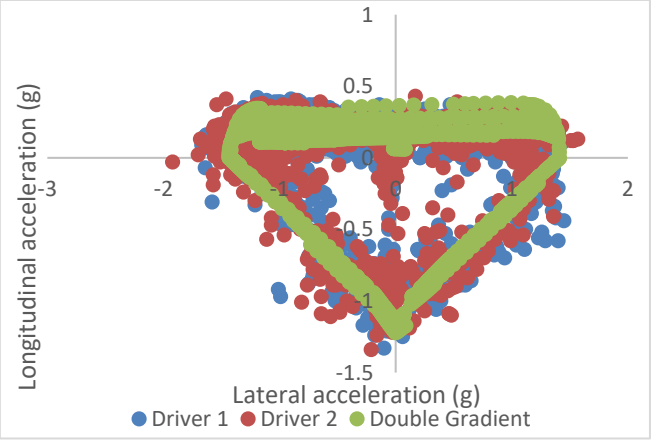

Fig. 8. Simulated g-g diagram compared to those obtained with professional drivers.

5 Conclusion

In this paper, a new optimization method for the trajectory optimization problem was presented, where control points are used to manipulate a spline representing the path of the vehicle. Since the solution to the problem involves direct manipulation of the

spline control points for path construction (see **Fig. 1**) and these control points are manipulated by a vector of linear interpolation parameters that vary between zero and one, the matrix product structure modulated by an activation function of Neural Networks was suitable for this application.

By changing the topology, the role of the Neural Network variables was changed as follows:

- The input variables have begun to act as reference vectors. For the optimization problem presented in this paper, the most suitable reference vector is the centerline of the track.
- The modifications applied to the network topology were made to emulate the specialization phenomenon of a Neural Network by adjusting the weights to determine the best path.
- The Neural Network output vector is the result of the problem posed to it. In the modified network, the output of the network is passed to the simulator, which calculates the vehicle lap time. The task of the network is now to adjust the vector of decision variables of the optimization problem, i.e. to change the layout of the center line of the race track until the lowest possible lap time is reached.
- The adaptation law of the Neural Network is constructed as a function of the error gradient. In the modified network, the adaptation law is constructed based on the gradients of the simulator outputs (lap time and lateral acceleration) and the gradient of the vector of decision variables. The adaptation law uses two functions where two gradients are applied, hence the name of the proposed method.

5.1 Algorithm limitations:

- Since it is based on gradients, long stretches of straight lines can produce a local minimum. This condition has a small effect on the total lap time, but it can produce a path that is unusual for professional drivers. Once drivers travel a known path in the straight lines, the control points on these sections can be set at the known positions, minimizing this problem.
- The algorithm assumes that the vehicle simulation model is able to follow the path defined by the spline. This is done either by linking the curvature and spline length data to the model equations or by running a control algorithm that follows the optimization trajectory.

5.2 Algorithm applications:

- Vehicle setup assistance during test sections in competitions.
- Baseline prediction for autonomous vehicle competitions.
- Development of amateur and semi-professional drivers, helping to learn the best race line for the circuit of interest.

5.3 Future works

- Generalize the method for solving various problems graphically by cubic splines or surfaces manipulation.
- Trajectory optimization in restricted 3D space such as racetracks or test tracks modeled in three dimensions.
- Trajectory optimization in unrestricted 3d space for applications with drones, Unmanned Aerial Vehicle (UAV) or Autonomous underwater vehicle (AUV).

– **Acknowledgement: The Authors would like to thank Bahia Research Foundation (FAPESB) for the financial support (grant 0342/2021).**

6 References

1. M. GADOLA, D. CAMBIAGHI, L. MANZO and D. VETTURI, “A Tool for Lap Time Simulation,” SAE Technical Paper 962529, Dezembro 1996.
2. S. V. KOUTRIK, “Optimal Control for Race Car Minimum Time Maneuvering,” Faculty of Mechanical, Maritime and Materials Engineering - Delft University of Technology, Delft, 2015.
3. D. CASANOVA, “On Minimum Time Vehicle Manoeuvring: The Theoretical Optimal Lap,” Cranfield University, School of Mechanical Engineering, 2000.
4. D. P. KELLY, “Lap Time Simulation with Transient Vehicle and Tyre Dynamics,” Cranfield University School of Engineering, Automotive Studies Group, 2008.
5. F. BIRAL and R. LOT, “A Curvilinear Abscissa Approach for the Lap Time Optimization of Racing Vehicles,” Cape Town, 2014.
6. N. D. BIANCO, R. LOT and M. GADOLA, “Minimum Time Optimal Control Simulation of a GP2 Race Car,” Journal of Automobile Engineering, vol. 232, nº 9, pp. 1180-1195, 2018.
7. T. HERRMANN, F. PASSIGATO, J. BETZ and M. LIENKAMP, “Minimum Race-Time Planning-Strategy for an Autonomous Electric Racecar,” 2020. [Online]. Available: arXiv:2005.07127.
8. S. LOVATO and M. MASSARO, “A Three-Dimensional Free-Trajectory Quasi-Steady-State Optimal-Control Method for Minimum-Lap-Time of Race Vehicles,” Vehicle System Dynamics, DOI: 10.1080/00423114.2021.1878242, 2021.
9. E. A. BASTOS, “Otimização de Seções Retangulares de Concreto Armado Submetidas à Flexo-Compressão Oblíqua Utilizando Algoritmos Genéticos,” Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2004.
10. A. A. CHAVES, “Heurísticas Híbridas com Busca Através de Agrupamentos para o Problema do Caxeiro Viajante com Coleta de Prêmios,” Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2005.
11. J. C. GROSS, “Derivative-Free Methods for High-Dimensional Optimization with Application to Centrifugal Pump Design,” University of Cambridge, Cambridge, 2021.
12. B. LENZO and V. ROSSI, “A Simple Mono-Dimensional Approach for Lap Time Optimisation,” Appl. Sci., vol. 1498, nº 10, February 2020.
13. A. JAIN and M. MORARI, “Computing the Racing Line Using Bayesian Optimization,” 12 February 2020.

14. S. SRINIVASAN, I. SA, A. ZYNER, V. REIJGWART, M. I. VALLS and R. SIEGWART, “End-to-End Velocity Estimation For Autonomous,” IEEE Robotics and Automation Letters., vol. 5, nº 4, pp. 6869-6875, Oct. 2020.
15. N. R. KAPANIA, J. SUBOSITS and J. C. GERDES, “A Sequential Two-Step Algorithm for Fast Generation of Vehicle Racing Trajectories.,” ASME. J. Dyn. Sys., Meas., Control, vol. 138, nº 9, September 2016.
16. S. GARLICK and A. BRADLEY, “Real-Time Optimal Trajectory Planning for Autonomous Vehicles and Lap Time Simulation Using Machine Learning,” 18 March 2021.
17. A. KONAR, Artificial Intelligence and Soft Computing: Behavioral and Cognitive Modeling of the Human Brain, Boca Raton: CRC Press, 1999.
18. J. L. R. FILHO, P. C. TRELEAVEN and C. ALIPPI, “Genetic Algorithm Programming Environments,” IEEE Computer Society Press, pp. 28-43, 1994.
19. R. STORN and K. PRICE, “Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces,” Journal of Global Optimization, nº 11, pp. 341-359, 1997.
20. F. N. MÓR, “An Evolutionary Approach for the Task Mapping Problem,” Pontifícia Universidade Católica do Rio Grande do Sul, Porto Alegre, 2016.
21. J. A. NELDER and R. MEAD, “A Simplex Method for Function Minimization,” The Computer Journal, vol. 7, pp. 308 - 313, 1965.
22. W. SPENDLEY, G. R. HEXT and F. R. HIMSWORTH, “Sequential Application of Simplex Designs in Optimisation and Evolutionary Operation,” Technometrics, vol. 4, p. 441, 1962.
23. F. GAO and L. HAN, “Implementing the Nelder-Mead simplex algorithm with adaptive parameters,” Comput Optim Appl, vol. 51, pp. 259-277, 2012.
24. A. RAVINDRAN, K. M. RAGSDELL and G. V. REKLAITIS, Engineering Optimization: Methods and Applications, 2ª ed., Hoboken, New Jersey: John Wiley and Sons, Inc., 2006.
25. C. M. GIULIANI, “Estratégias de Otimização não Diferenciável Aplicadas à Maximização da Produção de Campos de Petróleo,” Universidade Federal de Santa Catarina, Florianópolis, 2013.
26. A. R. CONN, N. I. M. GOULD and P. L. TOINT, Trust-Region Methods, Philadelphia: MPS/SIAM, 2000.
27. Y.-X. YUAN, “Recent Advances in Trust Region Algorithms,” Math. Program., vol. 151, pp. 249-281, 2015.
28. O. AFLAK, “Bézier Interpolation - Create smooth shapes using Bézier curves.,” 9 Maio 2020.
29. G. E. FARIN, Curves and Surfaces for Computer-Aided Geometric Design - A Practical Guide., 3ª ed., Arizona: Academic Press Inc., 2014.