Real-time Autonomous Parking in Unstructured Scenarios with an Indirect Optimal Control Approach

Edoardo Pagot[1], Mattia Piccinini[1], Alice Plebe[1], Enrico Bertolazzi[1], Francesco Biral[1]

***Abstract*— This paper presents a framework to generate in real-time autonomous parking maneuvers for generic and unstructured parking scenarios. Our method is based on the solution of minimum-time optimal control problems by means of an indirect approach. In a single optimization, the framework computes parking maneuvers composed of two or more segments of forward and reverse driving. The trajectory planning tasks are solved in real-time, and a fine grid is used to discretize the domain of the optimal control problems, resulting in accurate and collision-free solutions. Moreover, we introduce a novel method to deal with static obstacles in the optimal control problems, using penalty functions defined as regularized three-dimensional clip functions. The results show the effectiveness of our approach in various scenarios with narrow parking spots. A video demonstration made with the CARLA simulator is available at the following link: `https://youtu.be/Khob5QWEE-k`.**

I. INTRODUCTION

Current developments in intelligent vehicle technology focus on several aspects of autonomous driving, including autonomous parking and valet parking. Valet parking is the ability to autonomously cruise for a free parking spot, while autonomous parking refers more specifically to the execution of a parking maneuver. This paper focuses on the generation of autonomous parking maneuvers.

The expected benefits of autonomous parking and valet parking are several. There are technical advantages, such as the ability to move and park in narrower spaces compared to human-driven parking. Moreover, an autonomous system can find the fastest and shortest parking path, and it can minimize the search time for parking spot—which is anything but a minor matter. In the New York City region alone, vehicles cruising for a free parking space travel more than 70,000 miles every day. This equals to a daily emission of 29 metric tons of CO_2 [1]. Hence, autonomous parking systems can considerably impact greenhouse emissions. There are also advantages related to passenger comfort: for example, a passenger can exit the vehicle and let it look for a parking spot by itself.

To the best of the authors' knowledge, an autonomous framework able to plan in real-time complex parking maneuvers in generic scenarios (e.g., both parallel and reverse parking, or partially unstructured scenarios) is not present in the literature. Moreover, most of the optimal-control-based literature papers use direct collocation approaches, while indirect optimal control methods — based on the Pontryagin's Maximum Principle (PMP) — are still unexplored in the field of autonomous parking.

The contributions of this paper are the following.

1) We present a novel autonomous parking framework that solves minimum-time optimal control problems (OCPs) by means of an indirect method. The presented framework is able to deal with generic non-structured parking scenarios and narrow parking spots. In comparison with the more widespread direct collocation methods used in the literature, low computational times are obtained through the use of an indirect method and the software suite PINS. We use a dense time grid to discretize the domain of the OCP, which yields accurate solutions, also in the proximity of the obstacles.
2) The framework is able to compute complex parking maneuvers, composed of two or more forward and reverse driving segments in a single optimization, i.e. without the need for an iterative algorithm.
3) In order to define a generic parking environment, we devise a novel method to deal with static obstacles in the optimal control problems, using penalty functions described as smooth three-dimensional clip functions. The presented formulation allows us to create an arbitrary environment by combining multiple rectangular obstacles of different sizes.
4) We present a novel solution guess-generation scheme, combining a Hybrid A^* algorithm and an optimal-control tracking problem to provide robust guess solutions for the motion planning problem.

II. RELATED WORK

In recent years, several authors used optimal control (OC) to solve motion planning problems for autonomous parking. The authors of [2] used optimal control and direct collocation to compute minimum-time parking maneuvers in generic scenarios with multiple obstacles; however, they admitted that the computational times were far from real-time. The previous work was then extended in [3], in which a look-up table of pre-computed OC solutions was created and provided as guess for the minimum-time optimal control problem. The framework was successfully tested on narrow parking spots; however, only the parallel parking scenario was studied, and the computational times were in the order of 40-180 seconds, thus far-fetched for a real-time application. In [4], the authors obtained the parking maneuvers from the solution of a combined minimum-time and minimum-space optimal control problem. The OCP was initialized by means

[1]The authors are with the Dept. of Industrial Engineering, University of Trento, Trento, Italy (`name.surname@unitn.it`)

of standard maneuvers from a look-up table. They tuned and tested their framework on reverse parking only, and the computational times were in the order of 100 seconds. The authors of [5] presented an iterative framework for autonomous parking, based on the direct optimal control method. They reported promising real-time computational times lower than 1 second; however, the framework was validated on quite trivial scenarios, that were closer to a navigation problem than to a parking one. The authors of [6] developed a machine learning (ML) framework based on a Monte Carlo tree search, able to learn optimal parking maneuvers. The ML algorithm was trained with the solutions of minimum-time optimal control problems solved with direct collocation, which were computed offline; the computational time for the offline solution of the OCPs is not provided by the authors, and the framework was tested only on parallel parking scenarios. Finally, a remarkable application of optimal control for autonomous parking is [7], where the authors solved a minimum-time OCP with a convexification technique, in order to obtain an exact and smooth formulation of the original, non-convex OCP. They employed the Hybrid A^* algorithm [8] to compute solution guesses for the optimal control problem, obtaining average computational times lower than 2-3 seconds, and testing their framework on both reverse and parallel parking scenarios. Nevertheless, the parking spots were relatively large, and the resulting parking maneuvers were quite simple, being always composed of only two driving segments, the first in the forward direction, and the second in the rearward direction. Moreover, in order to reduce the computational time, the discretization grid of the OCP was quite coarse: as pointed out by the authors, some collisions with the obstacles are visible in the plotted OCP solutions, in the time frame between two consecutive discretization grid points.

III. METHODOLOGY

A. Vehicle Model

We adopt the following system of ordinary differential equations to model the vehicle motion:

$$\begin{aligned} \dot{x}(t) &= V(t)\cos(\theta(t)) \\ \dot{y}(t) &= V(t)\sin(\theta(t)) \\ \dot{\theta}(t) &= \frac{V(t)\tan(\delta(t))}{L} \\ \dot{V}(t) &= a(t) \\ \tau_\delta \dot{\delta}(t) &= \delta_{dot}(t), \end{aligned} \tag{1}$$

where $\{x, y, \theta\}$ are the cartesian coordinates of the vehicle reference point (center of the rear axle) and the orientation of the vehicle on the 2D plane, respectively. V is the vehicle forward velocity, while δ is the steering angle. The model states are $\boldsymbol{x} = \{x, y, \theta, V, \delta\}$, while the controls are $\boldsymbol{u} = \{a, \delta_{dot}\}$, where a is the vehicle forward acceleration and δ_{dot} is the steering velocity. Finally, L is the vehicle wheelbase. The notation $\dot{A}$ denotes the derivative of the quantity A with respect to the time t. The vehicle model is based on the geometry of a generic C-segment car, and it takes into account the front and rear overhangs from the wheel axles.

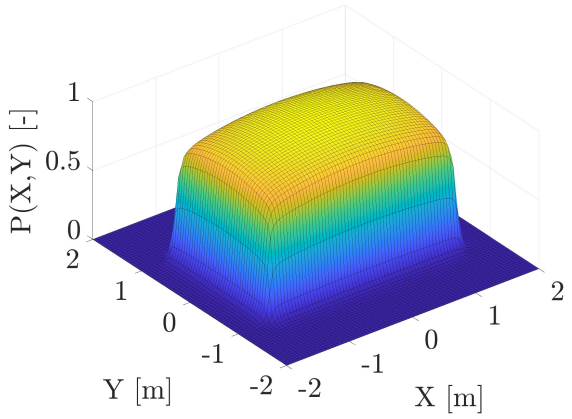


Fig. 1: Obstacle penalty: three-dimensional function based on smooth clip functions. The arguments of each penalty are the coordinates of a collision-control point. The value of the penalty is zero when the collision-control point is outside the obstacle bounding-box, and greater than zero when the collision-control point is inside the obstacle area.

B. Obstacle Formulation

The obstacles are taken into account in the optimal control problem by means of a novel penalty function formulation. Such a penalty function is built by multiplying two smooth clip functions, each one representing the size of the obstacle in the lateral and longitudinal direction:

$$P(x, y) = \text{Clip}(f_x(x, W, x_0))\ \text{Clip}(f_y(y, L, y_0)) \tag{2}$$

where $f_x(\cdot)$ and $f_y(\cdot)$ are two functions that enforce the chosen width W and length L of the obstacle, and its absolute position (x_0, y_0) in the 2D plane. A representation of a generic obstacle penalty function obtained with (2) is shown in Fig. 1. For collision checking, we define 12 points on the rectangular bounding box of the ego vehicle: the 4 corners of the box, 3 points equally spaced along the right and left sides of the vehicle, and the 2 middle-points of the front and rear edges of the box. We augment the cost function of the OCP (Section III-C) with the sum of the values of the penalty $P(x, y)$, evaluated in the (x, y) coordinates of each of the 12 points. The choice of 12 points for collision checking assures that the bounding box of the ego car does not collide with the obstacles at any point of its perimeter. Local collisions of the bounding box with obstacles could still happen on the box perimeter between control points; however, in our results we never encountered this event. Indeed, we started our experiments with only 4 control points: the number of 12 control points represents the lower bound for which we did not obtain collisions in the test scenarios employed in this paper.

C. Optimal Control Problem Formulation

We solve the following optimal control problem:

$$\min_{\boldsymbol{u}\in\mathcal{U}} \ \boldsymbol{J} = M\big(\boldsymbol{x}(T)\big) + \int_0^T l\big(\boldsymbol{x}(t),\boldsymbol{u}(t),t\big)\,\mathrm{d}t \tag{3}$$

$$\text{subject to } \ \dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t),\boldsymbol{u}(t)) \tag{4}$$

$$\boldsymbol{c}(\boldsymbol{x}(t),\boldsymbol{u}(t),t) \geq 0 \tag{5}$$

$$\boldsymbol{b}(\boldsymbol{x}(0),\boldsymbol{x}(T)) = 0, \tag{6}$$

where the dynamical system (4) is given by the system (1) transformed in the free-final-time formulation. The inequalities (5) implement box-constraints on the states and controls, such as limits on the maximum steering angle, forward velocity, acceleration and steering rate. The boundary conditions (6) constrain the vehicle pose at the starting and finish points, the latter being the target pose the car must reach at the end of the maneuver. Since we employ the indirect optimal control method, such boundary conditions are enforced strictly. In the cost function (3), the Mayer term $M\big(\,\cdot\,\big)$ contains the final time of the maneuver T_f, while the Lagrange term $l\big(\,\cdot\,\big)$ contains the summation of the penalty function (2) computed on the 12 collision-control points of the vehicle's bounding box perimeter.

The optimal control problem is formulated and solved using the software suite PINS [9], [10], [11], [12], [13]. Starting from the continuous-time OCP (3)-(6), PINS builds a two-point boundary value problem, which is then discretized. We select a time grid of 500 points for the discretization. The indirect optimal control method implemented by PINS and the proprietary solver allow us to solve the OCPs with small computations times on the fine mesh grid. The reader can refer to [14] and [15] for other recent examples of minimum-time OCPs solved in real-time using PINS.

D. Guess generation

The solution guess for the optimal control problem is generated by combining the Hybrid A^* algorithm [8] and an optimal control tracking problem. First, a parking maneuver is generated through the Hybrid A^* algorithm. The path provided by the Hybrid A^* is then tracked by solving a modified version of the OCP (3)-(6). The resulting solution is then used as a guess for the free-trajectory optimal control problem. The proposed guess generation method improves the robustness of the overall framework, leading to collision-free trajectories and decreasing the computational times.

IV. RESULTS

The presented autonomous parking framework is tested in three scenarios and with different starting positions, for a total of 63 maneuvers. The scenarios are defined as follows:

- Reverse parking scenario (RP): the maneuvers are solved on a grid of 28 starting positions. The distance between the two rows of obstacles is slightly larger than the vehicle length.
- Parallel parking scenario (PP): the maneuvers are solved on a grid of 21 starting positions. The maneuvering space on the right side of the vehicle is limited by a nearby obstacle.
- Generic parking scenario (GP): the maneuvers are solved on a grid of 14 starting positions. The vehicle must execute a reverse parking maneuver: the parking spot is quite distant and an obstacle is interposed between the starting and final positions.

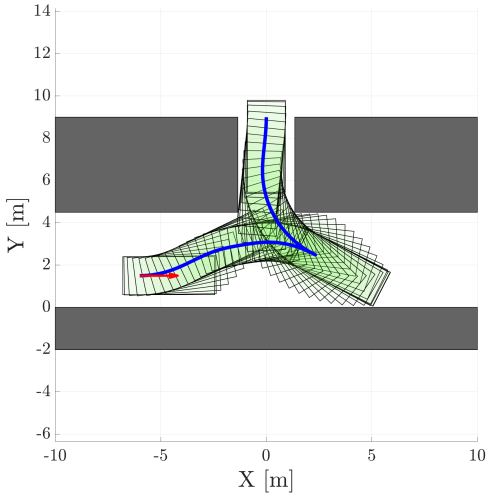
The optimal control problems are solved on a 2019 MacBook Pro equipped with a 2,6 GHz 6-Core Intel Core i7 processor. In Fig. 2, 3 and 4 we show the resulting maneuvers from two different starting points, in each parking scenario. The vehicle bounding box is plotted every 10 points of the discretization time grid employed in the OCP: notice that the vehicle exploits all the available space by navigating very close to the obstacles. The main factors that enable such accurate and real-time trajectory optimization are: the fine mesh with which the OCP is discretized, the use of a large number of collision-control points, the smooth formulation of the obstacle penalty functions (Section III-B), the effectiveness of the solution guess and of the PINS solver itself.

The presented autonomous parking framework is also able to compute complex maneuvers composed of three driving segments, as shown in Fig. 2b (rearward-forward-rearward segments) for the RP scenario and in Fig. 3a (forward-rearward-forward segments) for the PP scenario. A video demonstration of the six presented parking maneuvers was made with the CARLA driving simulator [16], and it can be found at the following link: `https://youtu.be/Khob5QWEE-k`.

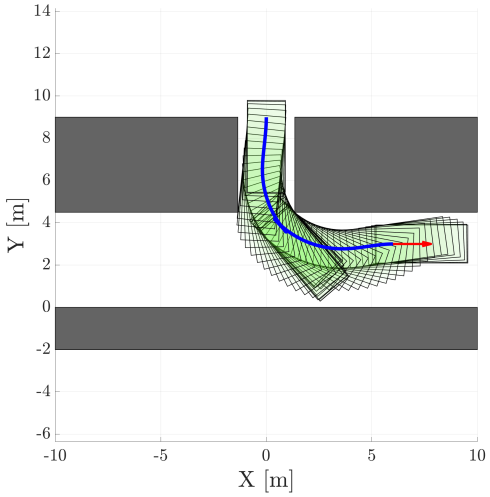
For the sake of brevity, the solutions of the other maneuvers are not plotted in this paper. Nevertheless, the computational times of the whole 63 parking maneuvers are reported in Fig. 5, subdivided by parking scenario. On average, the CPU time required to plan the parking maneuvers is around 0.6 seconds, and it is below 1 second for most maneuvers. The obtained results show that the presented framework is suitable for a real application, where the computational times should be low enough to avoid waiting for minutes before an automated parking maneuver is generated. Indeed, assuming an automotive-grade hardware 5 to 10 times slower than the hardware employed in this paper, the scaled-up computational times to plan the parking maneuvers would still be acceptable for a commercial application.

V. CONCLUSIONS

We presented a framework able to plan real-time autonomous parking maneuvers in generic and unstructured scenarios. The maneuvers are obtained as solutions of a minimum-time optimal control problem, which is solved by means of an indirect approach embedded in the software suite PINS. In order to improve the overall robustness of our framework and reduce the computation times, we also devised a novel approach to generate guess solutions for

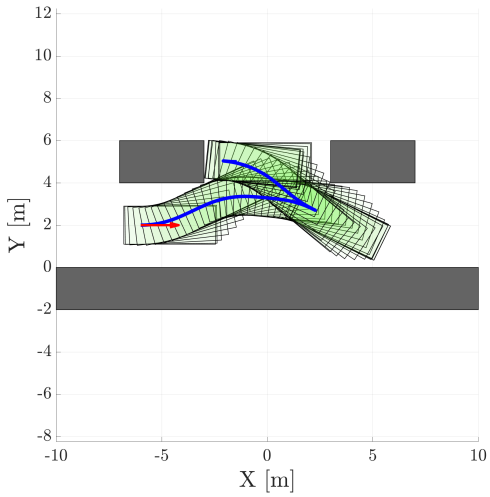


(a) Starting position before the target parking spot.

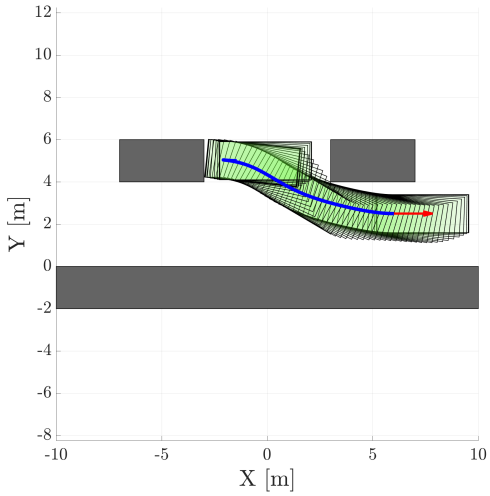


(b) Starting position after the target parking spot.

Fig. 2: Reverse parking scenario: the blue line represents the path of the vehicle reference point (middle point of the rear axle). The red arrow indicates the vehicle's forward direction at the starting point. Frames of the vehicle motion are shown every 10 discretization grid points of the OCP solution.



(a) Starting position before the target parking spot.



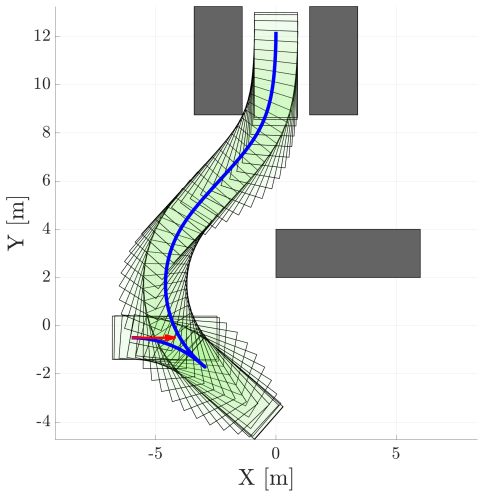
(b) Starting position after the target parking spot.

Fig. 3: Parallel parking scenario: the blue line represents the path of the vehicle reference point (middle point of the rear axle). The red arrow indicates the vehicle's forward direction at the starting point. Frames of the vehicle motion are shown every 10 discretization grid points of the OCP solution.

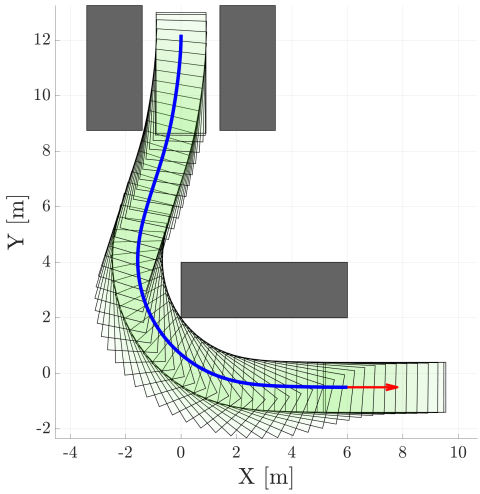
the optimal control problem. Obstacles penalties are written using a novel formulation based on smooth clip functions.

The results showed that the presented framework can deal with a wide range of maneuvers in different scenarios. The obtained solutions indicate that our method is able to navigate extremely close to the obstacles without collisions, thus exploiting all the available space to complete the parking maneuver. Moreover, we found that the presented system can generate complex maneuvers composed of two or more segments of forward and reverse driving. Finally, the computational times are significantly low and the framework is promising for a real-world automotive application.

Future work will be focused on the design of a low-level controller, able to track the OCP solutions to control a high-fidelity vehicle model in a simulation environment.

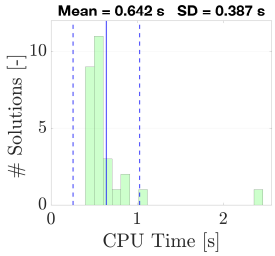


(a) Starting position before the obstacle.

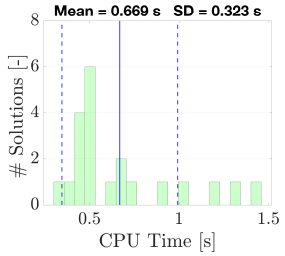


(b) Starting position after the obstacle.

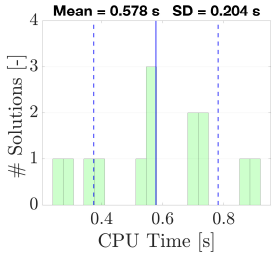
Fig. 4: Generic parking scenario: the blue line represents the path of the vehicle reference point (middle point of the rear axle). The red arrow indicates the vehicle's forward direction at the starting point. Frames of the vehicle motion are shown every 10 discretization grid points of the OCP solution.



(a) Reverse parking scenario.



(b) Parallel parking scenario.



(c) Generic scenario.

Fig. 5: Computational times on the three parking scenarios for different starting positions. The blue solid line indicates the mean computational time of each scenario, while the dashed lines indicate the standard deviation (SD). The numerical values of the computational times mean and SD are shown on top of each plot.

REFERENCES

[1] P. Ramaswamy, "Iot smart parking system for reducing green house gas emission," in *2016 international conference on recent trends in information technology (ICRTIT)*. IEEE, 2016, pp. 1–6.

[2] B. Li and Z. Shao, "A unified motion planning method for parking an autonomous vehicle in the presence of irregularly placed obstacles," *Knowledge-Based Systems*, vol. 86, pp. 11–20, 2015.

[3] B. Li, K. Wang, and Z. Shao, "Time-optimal maneuver planning in automatic parallel parking using a simultaneous dynamic optimization approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 11, pp. 3263–3274, 2016.

[4] C. Chen, B. Wu, L. Xuan, J. Chen, T. Wang, and L. Qian, "A trajectory planning method for autonomous valet parking via solving an optimal control problem," *Sensors*, vol. 20, no. 22, p. 6435, 2020.

[5] B. Li, T. Acarman, Y. Zhang, Y. Ouyang, C. Yaman, Q. Kong, X. Zhong, and X. Peng, "Optimization-based trajectory planning for autonomous parking with irregularly placed obstacles: A lightweight iterative framework," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[6] S. Song, H. Chen, H. Sun, M. Liu, and T. Xia, "Time-optimized online planning for parallel parking with nonlinear optimization and improved monte carlo tree search," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2226–2233, 2022.

[7] X. Zhang, A. Liniger, and F. Borrelli, "Optimization-based collision avoidance," *IEEE Transactions on Control Systems Technology*, vol. 29, no. 3, pp. 972–983, 2020.

[8] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Path planning for autonomous vehicles in unknown semi-structured environments," *The International Journal of Robotics Research*, vol. 29, no. 5, pp. 485–501, 2010.

[9] E. Bertolazzi, F. Biral, and M. Da Lio, "Symbolic–numeric indirect method for solving optimal control problems for large multibody systems," *Multibody System Dynamics*, vol. 13, no. 2, pp. 233–252, 2005. [Online]. Available: https://doi.org/10.1007/s11044-005-3987-4

[10] E. Bertolazzi, F. Biral, and M. Da Lio, "Symbolic-numeric efficient solution of optimal control problems for multibody systems," *Journal of Computational and Applied Mathematics*, vol. 185, no. 2, pp. 404–421, 2006, special Issue: International Workshop on the Technological Aspects of Mathematics. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0377042705001238

[11] E. Bertolazzi, F. Biral, and M. Da Lio, "Real-time motion planning for multibody systems: Real life application examples," *Multibody System Dynamics*, vol. 17, no. 2, pp. 119–139, 2007.

[12] F. Biral, E. Bertolazzi, and P. Bosetti, "Notes on numerical methods for solving optimal control problems," *IEEJ Journal of Industry Applications*, vol. 5, no. 2, pp. 154–166, 2016.

[13] N. D. Bianco, E. Bertolazzi, F. Biral, and M. Massaro, "Comparison of direct and indirect methods for minimum lap time optimal control problems," *Vehicle System Dynamics*, vol. 57, no. 5, pp. 665–696, 2019. [Online]. Available: https://doi.org/10.1080/00423114.2018.1480048

[14] M. Piccinini, M. Larcher, E. Pagot, D. Piscini, L. Pasquato, and F. Biral, "A predictive neural hierarchical framework for on-line time-optimal motion planning and control of black-box vehicle models," *Vehicle System Dynamics*, vol. 0, no. 0, pp. 1–28, 2022.

[15] E. Pagot, M. Piccinini, and F. Biral, "Real-time optimal control of an autonomous rc car with minimum-time maneuvers and a novel kineto-dynamical model," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 2390–2396.

[16] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, 2017, pp. 1–16.