A Nonlinear Model Predictive Control Strategy for Autonomous Racing of Scale Vehicles

Vittorio Cataffo[1], Giuseppe Silano[2], Luigi Iannelli[1], Vicenç Puig[3], and Luigi Glielmo[1]

***Abstract*—A Nonlinear Model Predictive Control (NMPC) strategy aimed at controlling a small-scale car model for autonomous racing competitions is presented in this paper. The proposed control strategy is concerned with minimizing the lap time while keeping the vehicle within track boundaries. The optimization problem considers both the vehicle's actuation limits and the lateral and longitudinal forces acting on the car modeled through the Pacejka's magic formula and a simple drivetrain model. Furthermore, the approach allows to safely race on a track populated by static obstacles generating collision-free trajectories and tracking them while enhancing the lap timing performance. Gazebo simulations using the F1/10 simulator showcase the feasibility and validity of the proposed control strategy. The code is released as open-source making it possible to replicate the obtained results.**

***Index Terms*—Nonlinear Model Predictive Control, Autonomous Racing, F1/10 simulator, Autonomous Vehicle Navigation.**

I. Introduction

Over the last years, the need to provide more affordable mobility and to reduce greenhouse gases from needless idling is creating a high expectation environment as a perfect enabler for Autonomous Vehicles (AVs) and applications of autonomous driving [1].

In an attempt to push the limits towards the development of new technologies, numerous competitions are organized and held in major international conferences. Above all, the F1/10 Autonomous Racing competition [2], [3] is one of the most popular; its name derives from the use of 1:10 scaled-down car models. Depending on the task objective, the problem is faced with different levels of detail and approximations [4], [5]. Several approaches have been proposed in the various editions of the competition [6]–[8]. However, when vehicle nonlinearities are excited, the control task is inevitably more demanding and this opens up new challenges to be solved.

The Model Predictive Control (MPC) approach has been proved to be a promising solution to control the car motion while complying with its dynamics and multiple heterogeneous constraints [9], [10]. Specifically, Nonlinear Model Predictive Control (NMPC) has resulted particularly suitable to control autonomous racing cars when their agility is essential for the particular application and must be exploited at the best [11], [12]. However, the intrinsic capability of the framework to embed physical constraints comes with a cost: the high-computational load required to solve the optimization problem.

Advances in the computational capabilities of modern computers and improvements in the algorithms efficiency [13], [14] have made it possible to manage such complexity along with the real-time requirements to solve these problems. Several software frameworks [15], [16] have been released over the years to facilitate modeling, control design, and simulation for a broad class of NMPC applications.

Various works have investigated NMPC strategies both as a trajectory generator [5], [17] and as a tracking controller [6], [8]. A common problem is the tracking of the lane center line while avoiding collisions with obstacles placed along the track. In most cases, NMPC is used in the outer loop of a cascaded architecture to provide a reference trajectory to an inner loop tracking controller [6], [17]. This approach allows to attain the tracking-lane objectives, but it can cause problems since the NMPC generator does not consider the limitations posed by the low-level controller [10]. As a consequence, the generated trajectory could violate the vehicle's actuator limits resulting in an unfeasible solution.

To overcome this limitation, NMPC can be used to combine trajectory generation, subject to obstacle avoidance constraints, and trajectory tracking, subject to actuation limits and track boundaries, in a single optimization problem [9], [11], [12]. The so-formulated problem allows to keep tracking of the lane center line, preventing critical configurations that could move the vehicle out of the limited-width track, while taking the actuator limitations into account.

Following this line of research, an NMPC architecture for lane center line tracking for autonomous racing car competitions is here proposed by considering for the first time, to the best of authors knowledge, the case of 1:10 scale racing cars. The optimization problem considers both vehicle dynamics constraints and physical actuation limits. The car dynamics are described by a bicycle model and longitudinal and lateral forces acting on the vehicle are modeled by a drivetrain model and the Pacejka's magic formula, respectively. An identification problem is set up to obtain the model parameters using experimental data collected in the F1/10 simulator [3]. The NMPC approach is

[1]V. Cataffo, L. Iannelli, and L. Glielmo are with the Department of Engineering, University of Sannio in Benevento, Benevento, Italy (email: {v.cataffo1, luigi.iannelli, glielmo}@unisannio.it).

[2]G. Silano is with Ricerca sul Sistema Enegertico (RSE) S.p.A., Department of Generation Technologies and Materials, Milan, Italy, and also with the Faculty of Electrical Engineering, Czech Technical University in Prague, Prague, Czech Republic (email: giuseppe.silano@fel.cvut.cz).

[3]V. Puig is with the Center of Supervision, Security and Automatic Control, Universitat Politècnica de Catalunya, Terrassa, Spain (email: vicenc.puig@upc.edu).

This work was partially funded by the ECSEL Joint Undertaking (JU) research and innovation programme COMP4DRONES under grant agreement no. 826610, and by the European Union's Horizon 2020 research and innovation programme AERIAL-CORE under grant agreement no. 871479.

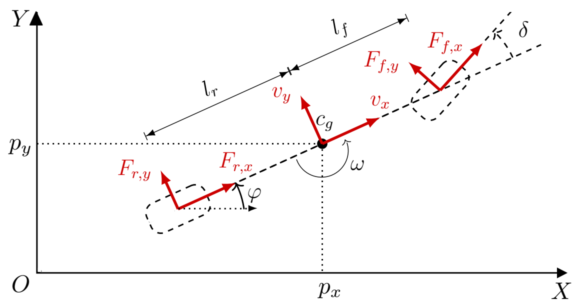

Figure 1: A representation of the dynamic bicycle model.

coded using the OpEn framework and PANOC as solver [15], [18]. Compared to the mentioned approaches, both the optimal racing trajectory and tracking lane problems are solved within the same optimization framework while avoiding static obstacles placed along the track. The code is released as open-source[1] making it possible to go through any part of the framework and to replicate the obtained results. Illustrative videos with the achieved Gazebo simulations are available at https://youtu.be/w5c328rQmX4.

II. System Modeling

A. Vehicle dynamics

The prediction model is a key part in MPC laws. One of the most common approaches used in vehicle dynamics applications is to simplify the vehicle model to that of a 2-wheeled bicycle model. This approximation is sufficient to provide the necessary inputs to actuators due to the small size of the vehicle. Similar approaches have been adopted by [6], [8], [12], [19] with different levels of detail and approximation.

Let us consider the bicycle model as described in [4], [19] and depicted in Fig. 1. The lateral and longitudinal forces $F_{r,y} \in \mathbb{R}$, $F_{r,x} \in \mathbb{R}$, $F_{f,y} \in \mathbb{R}$ and $F_{f,x} \in \mathbb{R}$ describe the forces acting on the tires, where the subscripts r and f refer to the rear and front parts of the vehicle, respectively, while the subscripts x and y denote the longitudinal (x) and lateral (y) axes along with the forces are exerted. The parameters $l_r \in \mathbb{R}_{\geq 0}$ and $l_f \in \mathbb{R}_{\geq 0}$ represent the distance between the rear and front wheel to the Center of Gravity (CoG) c_g, respectively. The steering angle $\delta \in \mathbb{R}$ quantifies the deflection of the front wheel, while the rear wheel is fixed. The vehicle's position and orientation in the *world frame* $\mathcal{F}_W$ are given by $p_x \in \mathbb{R}$, $p_y \in \mathbb{R}$ and $\varphi \in \mathbb{R}$, respectively.

The lateral forces $F_{f,y}$ and $F_{r,y}$ acting on the vehicle are described using the simplified Pacejka's magic formula [4], [19] as

$$F_{f,y} = D_f \sin\left(C_f \arctan\left(B_f \alpha_f\right)\right), \tag{1a}$$
$$F_{r,y} = D_r \sin\left(C_r \arctan\left(B_r \alpha_r\right)\right), \tag{1b}$$

where $B_f \in \mathbb{R}$ and $B_r \in \mathbb{R}$ are the *stiffness factors*, $C_f \in \mathbb{R}$ and $C_r \in \mathbb{R}$ are the *shape factors*, and $D_f \in \mathbb{R}$ and $D_r \in \mathbb{R}$ are the *peak factors*. Such an approximation allows meeting the trade-off between precision and computational requirements. The slip angles $\alpha_f \in \mathbb{R}$ and $\alpha_r \in \mathbb{R}$ are described as

$$\alpha_f = -\arctan\left(\frac{\omega l_f + v_y}{v_x}\right) + \delta, \tag{2a}$$
$$\alpha_r = \arctan\left(\frac{\omega l_r - v_y}{v_x}\right), \tag{2b}$$

where $\omega \in \mathbb{R}$ represents the change rate of the orientation φ during time. Equations (1) and (2) model the interaction between the car and the road.

For ease of modeling, the longitudinal forces $F_{f,x}$ and $F_{r,x}$ are assumed to be equal, i.e., $F_{f,x} = F_{r,x} = F_x$, and described using the following *drivetrain model* [19],

$$F_x = (C_{m1} - C_{m2} v_x)\tilde{d} - C_{m3} - C_{m4} v_x^2, \tag{3}$$

where $\tilde{d} \in [0, 1]$ is the Pulse Width Modulation (PWM) signal applied to motors and $C_{m1} \in \mathbb{R}_{\geq 0}$, $C_{m2} \in \mathbb{R}_{\geq 0}$, $C_{m3} \in \mathbb{R}_{\geq 0}$ and $C_{m4} \in \mathbb{R}_{\geq 0}$ are empirical parameters used to shape the model's response curve to fit the drivetrain characteristics [19]. The driving command $\tilde{d} = 1$ corresponds to full throttle, while $\tilde{d} = 0$ to full braking.

Hence, using Newton's second law, the vehicle dynamics with respect to (w.r.t.) c_g can be described as

$$\begin{cases} \dot{p}_x = v_x \cos\varphi - v_y \sin\varphi \\ \dot{p}_y = v_x \sin\varphi + v_y \cos\varphi \\ \dot{\varphi} = \omega \\ m\dot{v}_x = F_{r,x} - F_{f,y}\sin\delta + F_{f,x}\cos\delta + m v_y \omega \\ m\dot{v}_y = F_{r,y} + F_{f,y}\cos\delta + F_{f,x}\sin\delta - m v_x \omega \\ J_z \dot{\omega} = l_f F_{f,y}\cos\delta + l_f F_{f,x}\sin\delta - l_r F_{r,y} \end{cases}, \tag{4}$$

where $m \in \mathbb{R}_{>0}$ is the mass of the vehicle, $J_z \in \mathbb{R}_{>0}$ is the z-component of the inertia matrix $\mathbf{J} \in \mathbb{R}^{3\times 3}_{\geq 0}$, and v_x and v_y represent the car's velocity along the x-axis and y-axis of the body frame $\mathcal{F}_B$, respectively. Note that, with abuse of notation, the longitudinal forces $F_{f,x}$ and $F_{r,x}$ (3) are left in the model description (4) as the simplification $F_{f,x} = F_{r,x} = F_x$ does not affect the system modeling.

The model (4) describes a nonlinear dynamic system $\dot{\mathbf{x}} = f_c(\mathbf{x}, \mathbf{u})$, with state $\mathbf{x} = [p_x, p_y, \varphi, v_x, v_y, \omega]^\top \in \mathbb{R}^6$ and control input $\mathbf{u} = [\tilde{d}, \delta]^\top \in \mathbb{R}^2$.

B. System identification

In order to exploit the prediction model, it is pivotal to identify the model's parameters, i.e., those describing the lateral and longitudinal forces modeled by the simplified Pacejka's magic formula (1) and the drivetrain model (3). The set of parameters to be identified can be indicated as the vector $\boldsymbol{\zeta} \in \mathbb{R}^{10}$, and specifically $\boldsymbol{\zeta} = [B_f, B_r, C_f, C_r, D_f, D_r, C_{m1}, C_{m2}, C_{m3}, C_{m4}]^\top$. As for the remaining model's parameters, they are assumed to be known for the particular vehicle. Table I reports the list of parameters along with their values.

Acceleration and deceleration experiments in the F1/10 simulator can be used for the identification process based on a least squares minimization approach.

[1]https://bit.ly/3vPlmFf

Sym.	Value	Sym.	Value	Sym.	Value
l_f	$0.178\,\mathrm{m}$	l_r	$0.147\,\mathrm{m}$	m	$5.692\,\mathrm{kg}$
J_z	$0.204\,\mathrm{kg\,m^2}$	B_f	9.242	B_r	17.716
C_f	0.085	C_r	0.133	D_f	$134.585\,\mathrm{N}$
D_r	$159.919\,\mathrm{N}$	C_{m1}	$20\,\mathrm{N}$	C_{m2}	$6.92 \times 10^{-7}\,\mathrm{kg\,s^{-1}}$
C_{m3}	$3.99\,\mathrm{N}$	C_{m4}	$0.67\,\mathrm{kg\,m^{-1}}$	M	1674

Table I: Model's parameter values obtained through the identification procedure with signals acquired every $50\,\mathrm{ms}$.

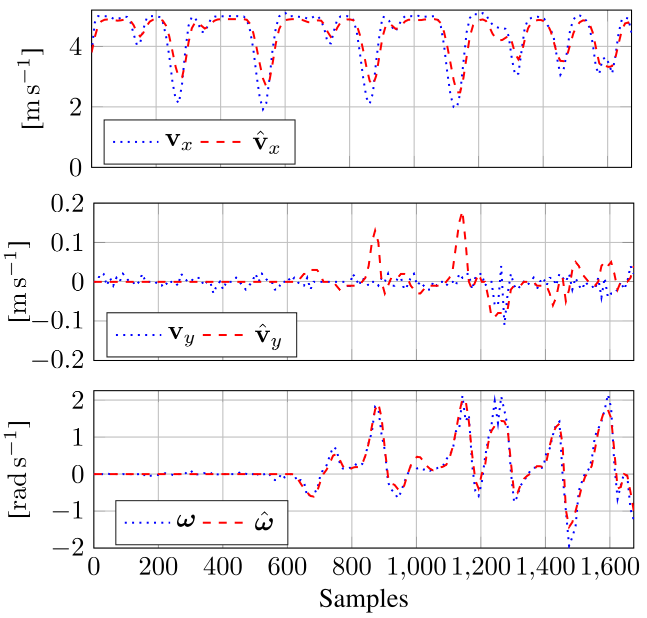

Figure 2: A comparison between the identified (hat symbols) and collected linear and angular velocity values.

By assuming having $M \in \mathbb{N}_{>0}$ signal samples acquired at a fixed sampling rate, the sequences of the linear velocities v_x and v_y and the angular velocity ω, can be denoted, respectively, by the vectors $\mathbf{v}_x = [v_{x_1}, v_{x_2}, \cdots, v_{x_M}]^\top$, $\mathbf{v}_y = [v_{y_1}, v_{y_2}, \cdots, v_{y_M}]^\top$, and $\boldsymbol{\omega} = [\omega_1, \omega_2, \cdots, \omega_M]^\top$. Thus, the least-squares minimization problem becomes

$$\min_{\underline{\boldsymbol{\zeta}} \leq \boldsymbol{\zeta} \leq \bar{\boldsymbol{\zeta}}} \|\mathbf{v}_x - \hat{\mathbf{v}}_x(\boldsymbol{\zeta})\|^2 + \|\mathbf{v}_y - \hat{\mathbf{v}}_y(\boldsymbol{\zeta})\|^2 + \|\boldsymbol{\omega} - \hat{\boldsymbol{\omega}}(\boldsymbol{\zeta})\|^2, \quad (5)$$

with $\hat{\mathbf{v}}_x(\boldsymbol{\zeta}) \in \mathbb{R}^M$, $\hat{\mathbf{v}}_y(\boldsymbol{\zeta}) \in \mathbb{R}^M$ and $\hat{\boldsymbol{\omega}}(\boldsymbol{\zeta}) \in \mathbb{R}^M$ representing the one-step prediction linear and angular velocities based on the discretized vehicle's model (4), while $\underline{\boldsymbol{\zeta}}$ and $\bar{\boldsymbol{\zeta}}$ denoting the range of minimum and maximum admissible values, respectively, for the parameters $\boldsymbol{\zeta}$.

The minimization problem (5) was coded using the 2019b release of MATLAB and solved using the `fmincon` function of the MathWorks Optimization Toolbox. Figure 2 shows the comparison between the acquired ($\mathbf{v}_x$, $\mathbf{v}_y$ and $\boldsymbol{\omega}$) and the predicted ($\hat{\mathbf{v}}_x$, $\hat{\mathbf{v}}_y$ and $\hat{\boldsymbol{\omega}}$) linear and angular velocities corresponding to the identified values of parameters.

III. Problem Formulation

A small-scale racing car is required to race along a track minimizing the lap time and remaining within the track boundaries. Meanwhile, the car is required to safely compete avoiding static obstacles populating the track. In addition, the control system is demanded to comply with the vehicle's actuation limits while fulfilling the mission objectives.

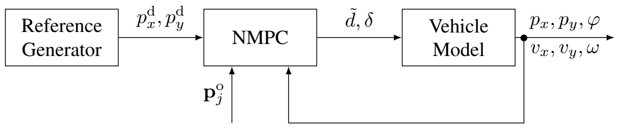

Figure 3: Block diagram of the proposed control strategy.

A two-layer control architecture has been proposed to cope with the problem of making a small-scale car race along a track minimizing the lap time and remaining within the track boundaries, by avoiding static obstacles, as well. A reference generator algorithm provides the reference point coordinates (p_x^d, p_y^d) to an NMPC tracking controller which computes the control signals ($\tilde{d}$, δ) to reach the target point while satisfying all constraints. Figure 3 describes the overall control system architecture.

The reference generator supplies reference point coordinates (p_x^d, p_y^d) to the NMPC tracking controller at each instance of the optimal problem. The planner leverages the capability of computing the projection of the car's position along the lane center line. For ease of experimentation, the proposed solution assumes to know the shape of the track, and therefore the lane center line. Such an assumption is in line with the competition rules [2] on which the algorithm was designed.

Let us assume the lane center line is sampled in $K \in \mathbb{N}_{>0}$ points with a constant sampling space (Euclidean distance) $d_s \in \mathbb{R}_{>0}$ and denote with $\mathbf{p}_k^\mathrm{c} = [p_{x_k}^\mathrm{c}, p_{y_k}^\mathrm{c}]^\top \in \mathbb{R}^2$ the k-th element of the lane center line, with $k \in \{1, 2, \ldots, K\}$. Hence, the lane center line can be represented as the sequence $\mathbf{p}^\mathrm{c} = \{\mathbf{p}_1^\mathrm{c}, \mathbf{p}_2^\mathrm{c}, \ldots, \mathbf{p}_K^\mathrm{c}\}$. Let us also define with $\mathbf{p} = [p_x, p_y]^\top \in \mathbb{R}^2$ the current car's position along the track. At each control step, the reference generator projects the car's position onto the center line:

$$i^* = \arg\min_i \|\mathbf{p}_i^\mathrm{c} - \mathbf{p}\|^2, \quad \begin{bmatrix} p_x' \\ p_y' \end{bmatrix} = \mathbf{p}_{i^*}^\mathrm{c}. \quad (6)$$

Afterwards, the planner computes the reference coordinates (p_x^d, p_y^d) looking-ahead P waypoints along the lane center line following a variation of the well-known pure-pursuit approach [20]. Figure 4 shows a schematic representation of the overall process for a single instance of the reference generator process.

Note that the projection operation and the reference coordinates computation are affected by the number of samples K and the number of look-ahead waypoints P. Those are key parameters that can be tuned to achieve the best performance and push the vehicle towards the limits trying, at the same time, not to increase the computation burden. Table II reports the values used for the numerical simulation along with the NMPC controller parameters.

The car is required to minimize the lap time while avoiding static obstacles placed along the track. Positions and size of the obstacles are assumed to be known in the whole

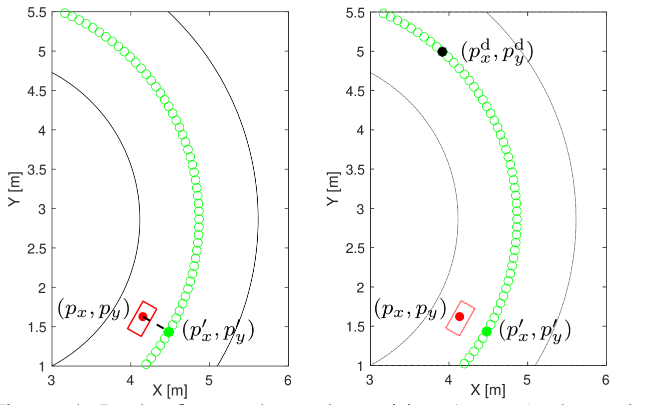

Figure 4: In the figure, the car's position $(p_x,\, p_y)$, the car's projection on the lane center line $(p'_x,\, p'_y)$ and the reference coordinate $(p^\mathrm{d}_x,\, p^\mathrm{d}_y)$ P samples looking-ahead w.r.t. the car's center lane projection.

Sym.	Value	Sym.	Value	Sym.	Value
P	90	T_s	0.033 s	N	50
R_c	0.24 m	R_g	2 m	Γ	1.5 m
$\mathbf{Q}_1$	diag(10, 10)	$\mathbf{Q}_2$	diag(10, 10)	R_j	1 m
$\underline{\gamma}_{v_x}$	0	$\bar{\gamma}_{v_x}$	5	d_s	0.1 m
$\underline{\boldsymbol{\mu}}$	$(0, -\pi/6)^\top$	$\bar{\boldsymbol{\mu}}$	$(1, \pi/6)^\top$	-	-

Table II: Control parameters and minimum ($\underline{\gamma}_{v_x}$) and maximum ($\bar{\gamma}_{v_x}$) admissible values of the state variable v_x.

prediction horizon of the NMPC. Besides, the shape of the obstacles is approximated with that of a circle of radius $R_j \in \mathbb{R}_{>0}$, where $j \in \{1, 2, \dots, O\}$, with $O \in \mathbb{N}$ the number of obstacles along the track. A schematic representation is depicted in Fig. 5.

The collision avoidance constraint is formulated through the positions of the vehicle $\mathbf{p}$ and the j-th obstacle $\mathbf{p}^\mathrm{o}_j$:

$$\|\mathbf{p} - \mathbf{p}^\mathrm{o}_j\|^2 \geq \Gamma_j^2, \tag{7}$$

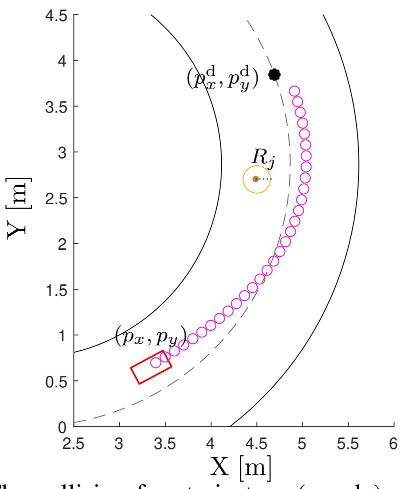

Figure 5: The collision-free trajectory (purple) computed by the NMPC tracking controller (a static obstacle in brown).

.

where $\Gamma_j \in \mathbb{R}_{>0}$ is a threshold distance value that the vehicle has to maintain to avoid collisions with obstacles. The latter is defined accounting for the obstacle sizes and the vehicle's body dimensions in order to ensure a sufficient room margin for maneuvers while remaining within the track boundaries.

Track boundaries constraints are also embedded into the optimal control formulation to maintain the vehicle within the limited-width track. By considering the vehicle's position $\mathbf{p} = [p_x, p_y]^\top$ and its projection on the lane center line $\mathbf{p}' = [p'_x, p'_y]^\top$, the constraint can be formulated as follows

$$\|\mathbf{p} - \mathbf{p}'\|^2 \leq (R_g - R_c)^2, \tag{8}$$

where R_g and R_c are defined considering the track width and the car dimension.

Thus, the optimal control problem, with a prediction horizon of $N \in \mathbb{N}_{>0}$ steps, is described as the minimization of the distance between the last predicted vehicle's position $\mathbf{p}_N$ and the reference point coordinates $\mathbf{p}^\mathrm{d}$. Therefore, at each time step $t_i = iT_s$, with $i \in \mathbb{N}_{>0}$ and T_s being the sampling time, it can be formulated an optimization problem as follows

$$\underset{\mathbf{u}}{\text{minimize}} \quad \|\mathbf{p}_N - \mathbf{p}^\mathrm{d}\|^2_{\mathbf{Q}_1} + \sum_{k=0}^{N-1} \|\mathbf{u}_k - \mathbf{u}_{k-1}\|^2_{\mathbf{Q}_2} \tag{9a}$$

s.t.

$$\mathbf{u}_{-1} = \mathbf{u}(t_{i-1}), \tag{9b}$$
$$\mathbf{x}_0 = \mathbf{x}(t_i), \tag{9c}$$
$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k),\ k = 0, \dots, N-1, \tag{9d}$$
$$\|\mathbf{p}_k - \mathbf{p}^\mathrm{o}_{j_k}\|^2 \geq \Gamma_j^2,\ k = 0, \dots, N,\ j \in \{1, \dots, O\}, \tag{9e}$$
$$\|\mathbf{p}_k - \mathbf{p}'_k\|^2 \leq (R_g - R_c)^2,\ k = 0, \dots, N-1, \tag{9f}$$
$$\underline{\boldsymbol{\mu}} \leq \mathbf{u}_k \leq \bar{\boldsymbol{\mu}},\ k = 0, \dots, N-1, \tag{9g}$$
$$\underline{\boldsymbol{\gamma}} \leq \mathbf{x}_k \leq \bar{\boldsymbol{\gamma}},\ k = 0, \dots, N, \tag{9h}$$

where (9a) is the objective function with $\mathbf{Q}_1, \mathbf{Q}_2 \in \mathbb{R}^{2\times 2}$ being diagonal weighting matrices, $\mathbf{x}_k$ and $\mathbf{u}_k$ are the sampled predicted state and control input, respectively, at the k-th sample of the current MPC interval, (9b) and (9c) initialize the control and the state, (9d) describes the discretized dynamic model for the vehicle (4), and (9g) and (9h) are the control input and state limits, respectively.

The problem (9) was encoded using a *single shooting implementation* with the track boundaries (8) and obstacle avoidance (7) constraints treated using the *augmented Lagrangian* and the *penalty method* approaches, respectively, following the constraints formulation of the OpEn framework[2] [15], [18]. The vehicle's dynamics were integrated using a Forward Euler integration method with a sampling time $T_s = 33\,\mathrm{ms}$. The prediction horizon considers $N = 50$ steps, which gives an ahead prediction of $1.65\,\mathrm{s}$.

IV. Simulation Results

To demonstrate the validity of the proposed control strategy, numerical simulations using the F1/10 simulator were

[2]https://alphaville.github.io/optimization-engine

Average Computation Time	Without Obs.	With Obs.
Scenario 1	$0.9\,\mathrm{ms}$	$1.2\,\mathrm{ms}$
Scenario 2	$1.0\,\mathrm{ms}$	$1.4\,\mathrm{ms}$
Scenario 3	$1.1\,\mathrm{ms}$	$1.4\,\mathrm{ms}$

Table III: Average computation time of the NMPC for different scenarios.

performed. The NMPC strategy was coded using the OpEn framework and PANOC as solver [15], [18]. All simulations were performed on a laptop with an i7-10750H processor ($2.60\,\mathrm{GHz}$) and 16GB of RAM running on Ubuntu 18.04 alongside the Melodic Morenia release of ROS. The control algorithm runs at $30\,\mathrm{Hz}$ sampling rate. Videos with the simulations are available at https://youtu.be/w5c328rQmX4, while the open-source code can be found at https://bit.ly/3vPlmFf.

Three different tracks in which the vehicle run in counter-clockwise direction were considered. Figure 6 shows the driven trajectories in the world frame along with the velocity profile encoded in gradient colors for each of them. In all simulated scenarios, the car's velocity touches the actuation limits imposed by the vehicle dynamics, maintaining high values for most of the track. Minimum velocity values can be seen in the most demanding stretches, where the minimum value is just under $3\,\mathrm{m\,s^{-1}}$. It can be seen how the car slows down in sharper turns and accelerates at the exit of the turns.

Figure 6 reports also the controller behavior in presence of obstacles (bottom graphs) showing how it adapts the car's motion to avoid collisions. Figure 7 shows the values of the control inputs $\mathbf{u} = [\tilde{d}, \delta]^\top$ for all the scenarios. As can be seen from the graphs, the control inputs remain within the boundaries (9g).

Figure 8 shows the histograms of the computation time per control step of the NMPC strategy. Low computation time per control step is also showcased when considering static obstacles along the track. Hence, these graphs show the capability of the proposed framework to compute the necessary control actions to drive the vehicle pushing the limits without violating the constraints.

Table III reports the average computation time for the NMPC. The minimum and the maximum average computation time obtained during the test runs are $0.9\,\mathrm{ms}$ and $1.4\,\mathrm{ms}$, respectively, such that the sampling time of $33\,\mathrm{ms}$ is never missed. The former is retrieved in the case without obstacles and with the shortest number of stretches, the latter refers to the last simulations where the vehicle is demanded to race on a winding track populated by obstacles.

V. Conclusions

In this paper, an NMPC strategy for autonomous racing of scale vehicles was presented. Numerical simulations performed in the F1/10 simulator demonstrated the validity of the proposed control strategy in a scenario quite close to real implementations. The proposed approach has shown low computation times to solve the optimization problem making it effective for complex control maneuvers, such as those in autonomous racing applications. Future work will include advanced path planning solutions to deal with uncertainties on the lane center line position and more challenging scenarios will be investigated, such as the combination of static and dynamic obstacles, in the direction of field experiments.

References

[1] D. Omeiza *et al.*, "Explanations in Autonomous Driving: A Survey," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–21, 2021.

[2] A. Agnihotri *et al.*, "Teaching Autonomous Systems at 1/10th-scale: Design of the F1/10 Racecar, Simulators and Curriculum," in *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, 2020, pp. 657–663.

[3] V. S. Babu *et al.*, "f1tenth.dev - An Open-source ROS based F1/10 Autonomous Racing Simulator," in *IEEE 16th International Conference on Automation Science and Engineering*, 2020, pp. 1614–1620.

[4] J. Kong *et al.*, "Kinematic and dynamic vehicle models for autonomous driving control design," in *IEEE Intelligent Vehicles Symposium*, 2015, pp. 1094–1099.

[5] E. Alcalá *et al.*, "LPV-MP planning for autonomous racing vehicles considering obstacles," *Robotics and Autonomous Systems*, vol. 124, p. 103392, 2020.

[6] ——, "Autonomous racing using Linear Parameter Varying-Model Predictive Control (LPV-MPC)," *Control Engineering Practice*, vol. 95, p. 104270, 2020.

[7] J. Klapálek *et al.*, "Car Racing Line Optimization with Genetic Algorithm using Approximate Homeomorphism," in *IEEE International Conference on Intelligent Robots and Systems*, 2021, pp. 601–607.

[8] R. Verschueren *et al.*, "Time-optimal race car driving using an online exact hessian based nonlinear MPC algorithm," in *European Control Conference*, 2016, pp. 141–147.

[9] U. Rosolia *et al.*, "Autonomous Vehicle Control: A Nonconvex Approach for Obstacle Avoidance," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 2, pp. 469–484, 2017.

[10] ——, "Learning How to Autonomously Race a Car: A Predictive Control Approach," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 6, pp. 2713–2719, 2020.

[11] H. Guo *et al.*, "Simultaneous Trajectory Planning and Tracking Using an MPC Method for Cyber-Physical Systems: A Case Study of Obstacle Avoidance for an Intelligent Vehicle," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 9, pp. 4273–4283, 2018.

[12] C. Guoying *et al.*, "Design and experimental evaluation of an efficient MPC-based lateral motion controller considering path preview for autonomous vehicles," *Control Engineering Practice*, p. 105164, 2022.

[13] R. Verschueren *et al.*, "acados—a modular open-source framework for fast embedded optimal control," *Mathematical Programming Computation*, vol. 11, pp. 1–37, 2021.

[14] J. A. E. Andersson *et al.*, "CasADi - A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, pp. 1–36, 2019.

[15] A. Sathya *et al.*, "Embedded nonlinear model predictive control for obstacle avoidance using PANOC," in *European Control Conference*, 2018, pp. 1523–1528.

[16] Y. Chen *et al.*, "MATMPC - A MATLAB Based Toolbox for Real-time Nonlinear Model Predictive Control," in *18th European Control Conference*, 2019, pp. 3365–3370.

[17] J. V. Frasch *et al.*, "An auto-generated nonlinear MPC algorithm for real-time obstacle avoidance of ground vehicles," in *European Control Conference*, 2013, pp. 4136–4141.

[18] S. Pantelis *et al.*, "OpEn: Code Generation for Embedded Nonconvex Optimization," in *IFAC-PapersOnLine*, vol. 53, no. 2, 2020, pp. 6548–6554.

[19] A. Liniger *et al.*, "Optimization-based autonomous racing of 1:43 scale RC cars," *Optimal Control Applications and Methods*, vol. 36, no. 5, pp. 628–647, 2014.

[20] H. Atoui *et al.*, "Real-Time Look-Ahead Distance Optimization for Smooth and Robust Steering Control of Autonomous Vehicles," in *29th Mediterranean Conference on Control and Automation*, 2021, pp. 924–929.

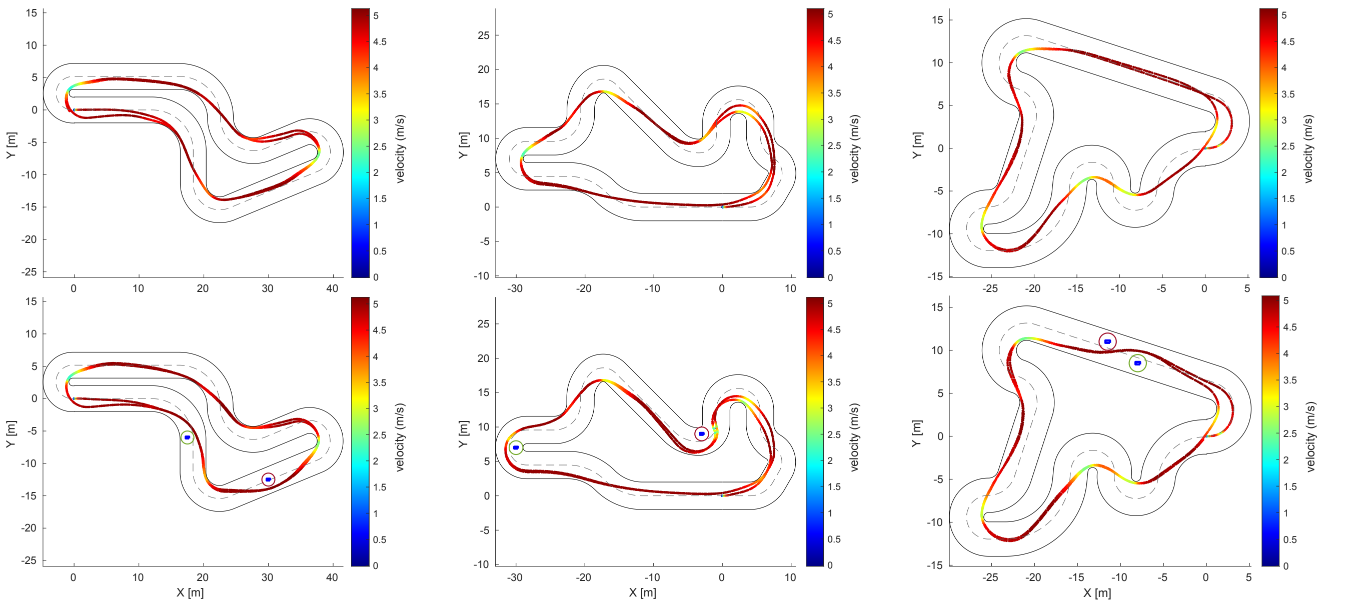
Figure 6: Simulation tracks. The color gradient indicates the car's velocity. Track scenarios with obstacles are at the bottom.

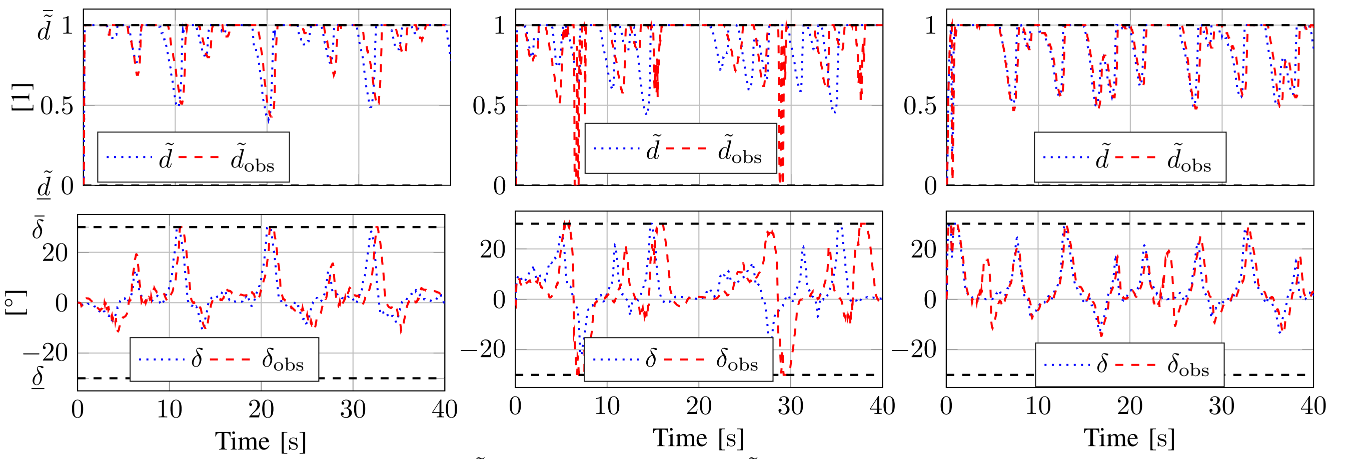
Figure 7: Control inputs $\mathbf{u}$ in cases with ($\tilde{d}_{\text{obs}}$, δ_{obs}) and without ($\tilde{d}$, δ) obstacles. The scenarios are in the same order as Fig. 6.

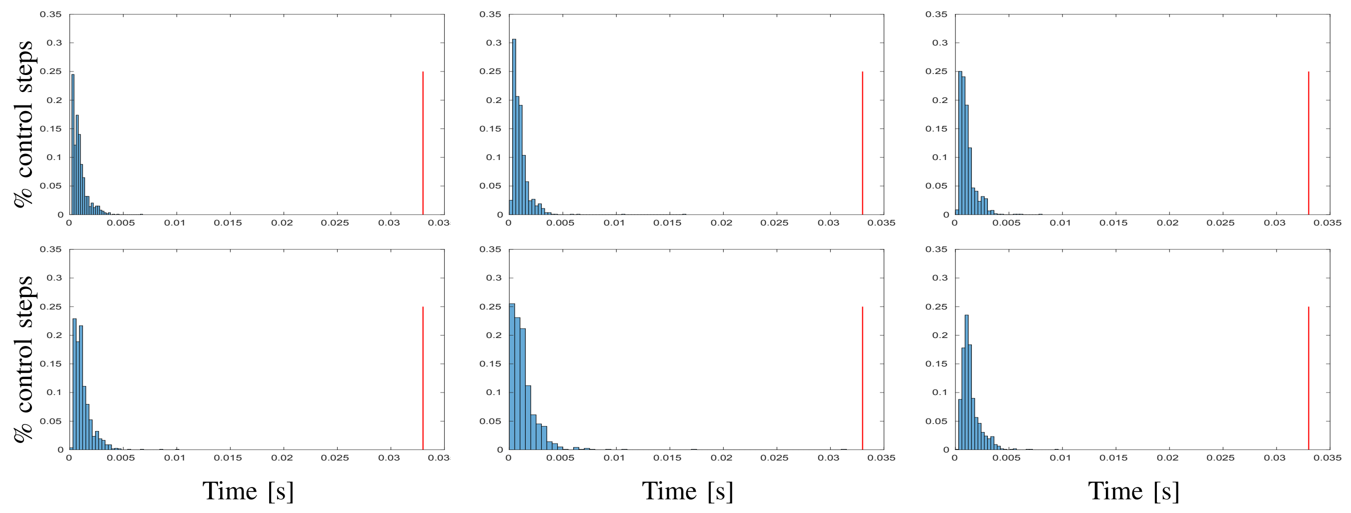
Figure 8: Histogram of the computation time of the NMPC step. The order follows that of Fig. 6. The red bar is $T_s = 33\,\text{ms}$.