Autonomous Racing with Multiple Vehicles using a Parallelized Optimization with Safety Guarantee using Control Barrier Functions

Suiyi He[1*], Jun Zeng[2*], and Koushil Sreenath[2]

***Abstract*— This paper presents a novel planning and control strategy for competing with multiple vehicles in a car racing scenario. The proposed racing strategy switches between two modes. When there are no surrounding vehicles, a learning-based model predictive control (MPC) trajectory planner is used to guarantee that the ego vehicle achieves better lap timing performance. When the ego vehicle is competing with other surrounding vehicles to overtake, an optimization-based planner generates multiple dynamically-feasible trajectories through parallel computation. Each trajectory is optimized under a MPC formulation with different homotopic Bezier-curve reference paths lying laterally between surrounding vehicles. The time-optimal trajectory among these different homotopic trajectories is selected and a low-level MPC controller with control barrier function constraints for obstacle avoidance is used to guarantee the system's safety-critical performance. The proposed algorithm has the capability to generate collision-free trajectories and track them while enhancing the lap timing performance with steady low computational complexity, outperforming existing approaches in both timing and performance for an autonomous racing environment. To demonstrate the performance of our racing strategy, we simulate with multiple randomly generated moving vehicles on the track and test the ego vehicle's overtaking maneuvers.**

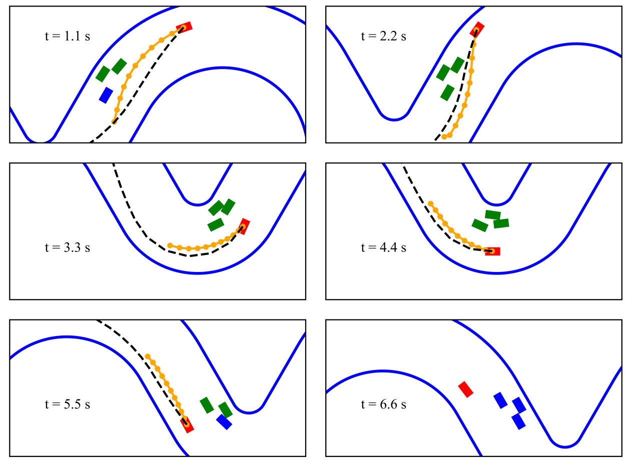

Fig. 1: Snapshots from simulation of the overtaking behavior. The red vehicle is the ego vehicle, the orange line shows the optimized trajectory from the planning strategy, and the dashed black line is the reference Bezier-curve used for generating the selected trajectory. Other vehicles are marked in blue with those in the ego vehicle's range of overtaking marked in green. The solid blue line is the track's boundary.

I. Introduction

A. Motivation

Recently, autonomous racing is an active subtopic in the field of autonomous driving research. In autonomous racing, the ego car is required to drive along a specific track with an aggressive behavior, such that it is capable of competing with other agents on the same track. By overtaking other leading vehicles and moving ahead, the ego vehicle can finish the racing competition with a smaller lap time. While the behavior of overtaking other vehicles has been studied in autonomous driving on public roads, however, these techniques are not effective on a race track. This is because autonomous vehicles are guided by dedicated lanes on public roads to succeed in lane follow and lane change behaviors, while the racing vehicles compete in the limited-width tracks without guidance from well-defined lanes. Existing work focuses on a variety of algorithms for autonomous racing, but most of them could not provide a time-optimal behavior with high update frequency in the presence of other moving agents on the race track. In order to generate racing behaviors for the ego racing car, we propose a racing algorithm for planning and control that enables the ego vehicle to maintain time-optimal maneuvers in the absence of local vehicles, and fast overtake maneuvers when local vehicles exist.

B. Related Work

In recent years, researchers have been focusing on planning and control for autonomous driving on public roads. For competitive scenarios like autonomous lane change or lane merge, both model-based methods [1] and learning-based methods [2] have been demonstrated to generate the ego vehicle's desired trajectory. Similarly, control using model-based methods [3]–[5] and learning-based methods [6] has also been developed. However, the criteria to evaluate planning and control performance are different for car racing compared to autonomous driving on public roads. For autonomous racing [7], when the ego racing car competes with other surrounding vehicles, most on-road traffic rules are not effective. Instead of maneuvers that offer a smooth and safe ride, aggressive maneuvers that push the vehicle to its dynamics limit [8] or even beyond its dynamics limit [9] are sought to win the race. In order to quickly overtake

*Authors have contributed equally and listed alphabetically.
[1]Author is with University of Minnesota-Twin Cities, MN 55455. he000231@umn.edu
[2]Authors are with University of California, Berkeley. {zengjunsjtu,koushils}@berkeley.edu

This work was partially supported through National Science Foundation Grant CMMI-1931853.

The animation video can be found at https://youtu.be/1zTXfzdQ8w4 and the implementation code can be found on https://github.com/HybridRobotics/car-racing.

The full version of this paper can be found at https://arxiv.org/abs/2112.06435.

surrounding vehicles, overtake maneuvers with tiny distances between the cars and large orientation changes are needed. Moreover, due to the bigger slip angle caused by changing the steering orientation more quickly during racing, more accurate dynamical models should be used for the design of planners and controllers for autonomous racing. We next enumerate the related work in several specific areas.

1) Planning Algorithms: For autonomous racing, the planner is desired to generate a time-optimal trajectory. Although some work using convex optimization problems [15], [22]–[27] or Bayesian optimization (BO) [28] reduces the ego vehicle's lap time impressively, either no obstacles [15], [23]–[28] or only static obstacles [22] are assumed to be on the track. When moving vehicles exist on the track, nonlinear dynamic programming (NLP) [19], graph-search [12] and game theory [13], [14], [29] based approaches have demonstrated their capabilities to generate collision-free trajectories. Additionally, in order to improve the chance of overtaking, offline policies are learnt for the overtake maneuvers at different portions of a specific track [30]. However, these approaches don't solve all challenges. For instance, work in [14], [19], [29], [30] does not take lap timing enhancement into account. In [12], the ego vehicle is assumed to compete on a straight track with one constant-speed surrounding vehicle. These assumptions are relatively simple for a real car racing competition. In [13], it is assumed that the planner knows the other vehicle's strategy and the complexity of the planner increases excessively when multiple vehicles compete with each other on the track.

2) Control Algorithms: Researchers focus on enhancing performance of the ego vehicle by achieving its speed and steering limits through better control design, e.g., obtaining the optimal lap time by driving fast. The majority of existing work focuses on developing controllers with no other vehicles on the track. The learning-based controllers [16]–[18], [31] leverage the control input bounds to achieve optimal performance in iterative tasks. Model-free methods like Bayesian optimization (BO) [32], Gaussian processes (GPs) [10], deep neural networks (DNN) [33], [34] and deep reinforcement learning (DRL) [11], [35] have also been exploited to develop controllers that result in agile maneuvers for the ego car. To deal with other surrounding vehicles, DRL has also been used in [36] to control the ego vehicle during overtake maneuvers. Recently, model predictive based controllers (MPC) with nonlinear obstacle avoidance constraints have become popular to help the ego vehicle avoid other vehicles in the free space. A nonconvex nonlinear optimization based controller is implemented in [37] to help the ego vehicle avoid static obstacles. Researchers in [38] use mixed-integer quadratic programs (MIQP) to help the ego vehicle compete with one moving vehicle. In [20], GPs was applied to formulate the distance constraints of a stochastic MPC controller with a kinematic bicycle model. However, large slip angles under aggressive maneuvers will cause a mismatch between the real dynamics model and the kinematic model used in the controller, resulting in the controller being unable to guarantee the system's safety in some cases. In [21], a safety-critical control design by using control barrier functions is proposed to generate a collision-free trajectory without a high-level planner, where infeasibility could arise due to the high nonlinearity of the optimization problem. Moreover, due to the lack of a trajectory planner, deadlock could happen very often during overtake maneuvers, such as in [20], [21]. A comparison of various approaches and their features are enumerated in TABLE I.

As mentioned above, all the previous work on planning and control design for autonomous racing could not enhance the lap timing performance and simultaneously compete with multiple vehicles. Inspired by the work on iterative learning-based control and optimization-based planning, we propose a novel racing strategy to resolve the challenges mentioned above with a steady low computational complexity.

C. Contribution

The contributions of this paper are as follows:

- We present an autonomous racing strategy that switches between a learning-based MPC trajectory planner (in the absence of surrounding vehicles) and an optimization-based homotopic trajectory planner with a low-level safety-critical controller (when the ego vehicle competes with surrounding vehicles).
- The learning-based MPC approach guarantees time-optimal performance in the absence of surrounding vehicles. When the ego vehicle competes with surrounding vehicles, multiple homotopic trajectories are optimized in parallel with different geometric reference paths and the best time-optimal trajectory is selected to be tracked with an optimization-based controller with obstacle avoidance constraints.
- We validate the robust performance together with steady low computational complexity of our racing strategy in numerical simulations where randomly moving vehicles are generated on a simulated race track. It is shown that our proposed strategy allows the ego vehicle to succeed in overtaking tasks without deadlock when there are multiple vehicles moving around the ego vehicle. We also demonstrate that our strategy would work for various racing environments.

II. Background

In this section, we revisit the vehicle model and learning-based MPC for iterative tasks. The learning-based MPC will be used as the trajectory planner when no surrounding vehicles exist.

A. Vehicle Model

In this work, we use a dynamic bicycle model with decoupled Pacejka tire model under Frenet coordinates. The system dynamics is described as follows,

$$\dot{x} = f(x, u), \tag{1}$$

Approach	*GP*	*DRL*	*Graph-Search*	*Game Theory*		*Model-Based*							
Publication	[10]	[11]	[12]	[13]	[14]	[15]	[16]	[17]	[18]	[19]	[20]	[21]	**Ours**
Lap Timing	**Yes**	**Yes**	**Yes**	**Yes**	No	**Yes**	**Yes**	**Yes**	**Yes**	No	No	No	**Yes**
Static Agent	No	No	**Yes**	**Yes**	**Yes**	No	No	No	No	**Yes**	**Yes**	**Yes**	**Yes**
Moving Agent	No	No	One	One	**Multiple**	No	No	No	No	**Multiple**	One	One	**Multiple**
Update Frequency (Hz)	N/A	N/A	15	**30**	2	Offline	**20**	**20**	Offline	$<$1	10	$<$10	$>$**25**
Planner	No	No	**Yes**	**Yes**	**Yes**	**Yes**	No	No	No	**Yes**	**Yes**	No	**Yes**
Dynamics Accuracy	N/A	N/A	**Yes**	**Yes**	No	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**

TABLE I: A comparison of recent work on autonomous racing and their attributes. Lap timing indicates if the lap timing performance is considered. Static and moving agent indicates if other static or moving agents are considered. Update frequency indicates the optimization update frequency. Learning-based approaches like GP don't have this attribute. Planner indicates if the approach has the planning part. Dynamics accuracy indicates the dynamics model used in the controller, with "Yes", "No", "N/A" representing dynamic model, kinematic model and model-free.

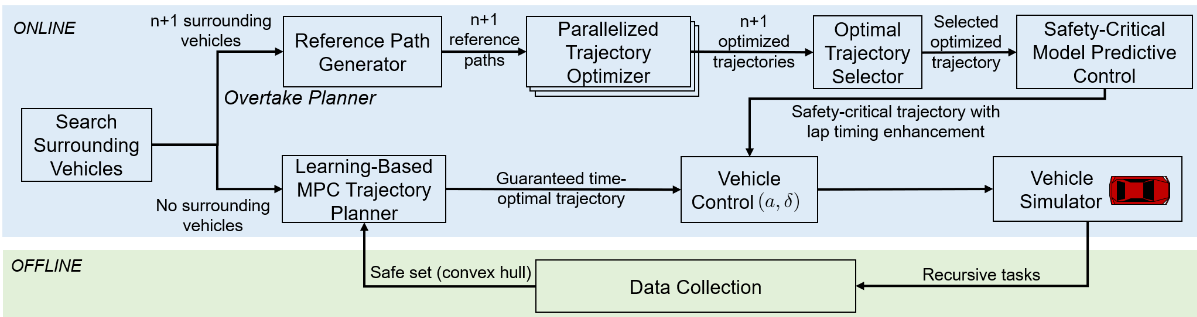

Fig. 2: Autonomous Racing Strategy: The system dynamics is identified through offline data collection via recursive tasks. For online deployment, when no surrounding vehicles exist, the learning-based MPC trajectory planner is executed to guarantee time-optimal trajectories. When there are surrounding vehicles, the best time-optimal trajectory is chosen among the $n+1$ trajectories that are optimized in parallel with each optimization carried out for a particular homotopic trajectory around the n surrounding cars. The chosen trajectory is then tracked with a safety-critical model predictive based controller.

where x and u show the state and input of the vehicle, f is a nonlinear dynamic bicycle model in [39]. The definition of state and input is as follows,

$$x = [v_x, v_y, \omega_z, e_\psi, s_c, e_y]^T, \ u = [a, \delta]^T, \tag{2}$$

where acceleration at vehicle's center of gravity a and steering angle δ are the system's inputs. The curvilinear distance travelled along the track's center line is denoted by s_c, while e_y and e_ψ show the deviation distance and heading angle error between vehicle and the track's center line. The longitudinal velocity, lateral velocity and yaw rate, are respectively denoted by v_x, v_y and ω_z.

In this paper, this model (1) is applied for precise numerical simulation using Euler discretization with sampling time 0.001s (1000Hz). Through linear regression from the simulated reference path, an affine time-invariant model

$$x_{t+1} = A(\bar{x})x_t + B(\bar{x})u_t \tag{3}$$

will be used in the trajectory planner to avoid excessive complexity from nonlinear optimization, where $\bar{x}$ represents the equilibrium point for the linearized dynamics. On the other hand, an affine time-varying model

$$x_{t+1} = A_t(\bar{x}_k)x_t + B_t(\bar{x}_k)u_t + C_t(\bar{x}_k), \tag{4}$$

is obtained, where matrices $A_t(\bar{x}_k)$, $B_t(\bar{x}_k)$, and $C_t(\bar{x}_k)$ are obtained at the local equilibrium point $\bar{x}_k$ on reference trajectory with iterative data which is close to x_t. This dynamics (4) will be used for the racing controller design for better tracking performance.

B. Iterative Learning Control

A learning-based MPC [16], which improves the ego vehicle's lap timing performance through iterative tasks, will be used in this paper. This has the following components:

1) Data Collection: The learning-based MPC optimizes the lap timing through historical states and inputs from iterative tasks. To collect initial data, a simple tracking controller like PID or MPC can be used for the first several laps. During the data collection process, after the j-th iteration (lap), the controller will store the ego vehicle's closed-loop states and inputs as vectors. Meanwhile, through offline calculation, every point of this iteration will be associated with a cost, which describes the time to finish the lap from this point.

2) Online Optimization: After the initial laps, the learning-based MPC optimizes the vehicle's behavior based on collected data. At each time step, the terminal constraint is formulated as a convex set. This convex set includes the states that can drive the ego vehicle to the finish line in the previous laps. By constructing the cost function to create a minimum-time problem, an open-loop optimized trajectory can be generated. Since the cost function is based on the previous states' timing data, the vehicle is able to drive to the finish line with time that is no greater than the time from the same position during previous laps. As a result, the ego vehicle will reach the time-optimal performance after several laps.

More details of this method can be found in [16]. In our work, this approach will be used for trajectory planning when the ego vehicle has no surrounding vehicles. This helps with better lap timing in the absence of surrounding vehicles. Notice that the data for iterative learning control will be collected through offline simulation with no obstacles on the track, as shown in Fig. 2.

III. Racing Algorithm

Having introduced the background on vehicle modeling and learning-based MPC, we will next present an autonomous racing strategy that can help the ego vehicle enhance lap timing performance while overtaking other moving vehicles.

A. Autonomous Racing Strategy

There are two tasks in autonomous racing: enhancing the lap timing performance and competing with other vehicles. To deal with these two tasks, our proposed strategy will switch between two different planning strategies. When there are no surrounding vehicles, trajectory planning with learning-based MPC is used to enhance the timing performance through historical data. When there are surrounding vehicles, an optimization-based trajectory planner optimizes several homotopic trajectories in parallel around the surrounding vehicles and a collision-free time-optimal trajectory is selected among these, which is then tracked by a low-level MPC controller. By adding obstacle avoidance constraints to the low-level controller, it has the ability to guarantee the system's safety. The racing strategy is summarized in Fig. 2. More details can be found in the full version [40].

B. Overtaking Planner

To determine if a surrounding vehicle is in the ego vehicle's range of overtaking, following condition must be satisfied:

$$-\epsilon l \leq s_{c,i} - s_c \leq \epsilon l + \gamma |v_x - v_{x,i}| \tag{5}$$

where s_c and $s_{c,i}$ are ego vehicle's and i-th surrounding vehicle's current traveling distance, v_x and $v_{x,i}$ are ego vehicle's and i-th surrounding vehicle's longitudinal speed. l indicates the vehicle's length. ϵ and γ are safety-margin factor and prediction factor which we can tune for different performance.

As shown in Fig. 3, when there are n vehicles in the ego vehicle's range of overtaking, there exists $(n+1)$ potential areas with each area leading to paths with a different homotopy that the ego vehicle can use to overtake these surrounding vehicles. These $n+1$ areas are the one below the n-th vehicle, the one above the 1st vehicle, and the ones between each group of adjacent vehicles. We then solve $n+1$ groups of optimization-based trajectory planning problems in parallel, enabling steady low computational complexity even when competing with different numbers of surrounding vehicles. To reduce each optimization problem's computational complexity, geometric paths with a distinct homotopy class that laterally lay between vehicles or vehicle and track boundary (black dashed curves in Fig. 3) are used as reference paths in the optimization problems. By comparing the optimization problems' costs, the optimal trajectory is selected from the

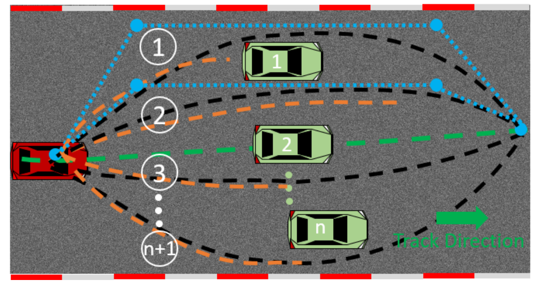

Fig. 3: A typical overtaking scenario when there are n vehicles in the range of overtaking. The ego vehicle and surrounding vehicles are in red and green, respectively. The dashed green line is a time-optimal trajectory calculated from the learning-based MPC. Blue points are control points for Bezier-curves in two different cases. Dashed black lines are $n+1$ groups of reference paths each with a different homotopy. Dashed orange lines are optimized trajectories for each optimization problem. The trajectory in area 2 is selected as the best for its smoothness and reachability along the track.

$n+1$ optimized solutions. For example, as the case shown in Fig. 3, the dashed orange line in area 2 will be selected since it avoids surrounding vehicles and finishes the overtake maneuver with a smaller time. The function to minimize during the selection is shown as follows,

$$\begin{aligned} J_s(x_t) = \min_{x_t} -K_s(s_{c_{t+N}} - s_{c_t}) - \sum_{k=1}^{N_p}((s_{c_{t+k}} \\ - s_{c,i_{t+k}})^2 + (e_{y_{t+k}} - e_{y,i_{t+k}})^2 - l^2 - d^2) + b \end{aligned} \tag{6}$$

where K_s is a scalar used in metric for timing and b is a non-zero penalty cost if the new potential area of overtaking is different from the area of overtaking in the last time step. A bigger value of K_s is applied such that the ego vehicle is optimized to reach a farther point during the overtake maneuver, which results in a shorter overtaking time since the planner's prediction horizon and sampling time are fixed. Additionally, the other terms in (6) prevents the ego vehicle from changing direction abruptly during an overtake maneuver and guarantees the ego vehicle's safety.

Bezier-curves are widely used in path planning algorithms in autonomous driving research [41]–[43] because it is easy to tune and formulate. Third-order Bezier-curves are used in this work. Each Bezier-curve is interpolated from four control points, including shared start and end points with two additional intermediate points, shown in Fig. 3. Specifically, the start point for the Bezier curve is the ego vehicle's current position and end point is on the time-optimal trajectory generated from learning-based MPC planner. The selection of the end point makes the vehicle's state as close as possible to the time-optimal trajectory after the overtake behavior. To make all curves smoother and have no or fewer conflicts with surrounding vehicles, the other two control points will be between the track's boundary and the vehicle for Areas 1 and n+1, or between two adjacent vehicles as shown in Fig. 3.These two intermediate control points will have the same lateral deviation from the center line. The key advantage of our selection of control points is that the interpolated geometric curve won't cross the connected lines between control points with its convexity, as shown in Fig. 3. This property makes our reference paths collision-free with respect to the surrounding vehicles in most cases, which speeds up the computational time of the trajectory generation in each area.

The details of the optimization formulation for trajectory generation will be illustrated in the next section.

C. Trajectory Generation

After illustrating the planing strategy, this subsection will present details about the optimization problem used for trajectory generation for each potential area with different homotopic paths that the ego vehicle can use to overtake the surrounding vehicles.

The optimization problem is formulated as follows,

$$\begin{aligned}
\underset{x_{t:t+N_p|t}, u_{t:t+N_p-1|t}}{\operatorname{argmin}} \quad & p(x_{t+N|t}) + \sum_{k=0}^{N_p-1} q(x_{t+k|t}) \\
& + \sum_{k=1}^{N_p-1} r(x_{t+k|t}, u_{t+k|t}, x_{t+k-1|t}, u_{t+k-1|t}) && \text{(7a)} \\
\text{s.t. } & x_{t+k+1|t} = Ax_{t+k|t} + Bu_{t+k|t}, k=0,...,N_p-1, && \text{(7b)} \\
& x_{t+k+1|t} \in \mathcal{X}, u_{t+k|t} \in \mathcal{U}, k=0,...,N_p-1, && \text{(7c)} \\
& x_{t|t} = x_t, && \text{(7d)} \\
& g(x_{t+k+1|t}) \geq d + \epsilon, k=0,...,N_p-1, && \text{(7e)}
\end{aligned}$$

where (7b), (7c), (7d) are constraints for system dynamics, state/input bounds and initial condition. The system dynamics constraint describes the affine linearized model described in (3). The cost function (7a) is composed of three parts along the horizon of length N_p, the terminal cost $p(x_{t+N|t})$, the stage cost $q(x_{t+k|t})$ and the state/input changing rate cost $r(x_{t+k|t}, u_{t+k|t})$. The construction of cost function and constraints in the optimization will be presented in detail in the following subsections.

1) Terminal Cost: The terminal cost is related to the ego vehicle's traveling distance along the track during the overtaking process, and is given by

$$p(x_{t+N|t}) = K_d(s_{c_{t+N|t}} - s_{c_t}). \tag{8}$$

This compares the open-loop predicted traveling distance at the N-th step $s_{c_{t+N|t}}$ with the ego vehicle's current traveling distance s_{c_t}. This works as the cost metric for timing during the overtaking process.

2) Stage Cost: The stage cost introduces the lateral position differences between the open-loop predicted trajectory and other two paths along the horizon,

$$q(x_{t+k|t}) = ||x_{t+k|t} - x_R(s_{c_k})||^2_{Q_1} + ||x_{t+k|t} - x_T(s_{c_k})||^2_{Q_2}, \tag{9}$$

where x_R and x_T are the reference path and time-optimal trajectories respectively in Frenet coordinates. The reference path x_R is the Bezier-curve in the corresponding area. The time-optimal trajectory x_T is generated by the learning-based MPC trajectory planner used on a track without other agents, as discussed in Sec. II-B. Note that s_{c_k} is an initial guess for the traveling distance at the k-th step, which is equal to $s_{c_k} = s_{c_t} + v_{x_t} k\Delta_t$, where a constant longitudinal speed is assumed along the prediction horizon.

3) State/Input Changing Rate Cost: To make the predicted trajectory smoother, the state/input changing rate cost $r(x_{t+k|t}, u_{t+k|t})$ is formulated as follows:

$$\begin{aligned}
& r(x_{t+k|t}, u_{t+k|t}, x_{t+k-1|t}, u_{t+k-1|t}) \\
=& ||x_{t+k|t} - x_{t+k-1|t}||^2_{R_1} + ||u_{t+k|t} - u_{t+k-1|t}||^2_{R_2}.
\end{aligned} \tag{10}$$

4) Obstacle Avoidance Constraint: In order to generate a collision-free trajectory, collision avoidance constraint (7e) is added in the optimization problem. To reduce computational complexity, only linear lateral position constraint will be added when the ego vehicle overlaps with other vehicles longitudinally. The inequality $|s_c(t) + v_x(t)k\Delta t - s_{c,i}(t+k)| < l + \epsilon$ will be used to check if the ego vehicle is overlapping with other vehicles longitudinally along the horizon. In (7e), $g(x) = |e_{y,i} - e_y|$ shows the lateral position difference, l and d are the vehicle's length and width, and ϵ is a safe margin.

After parallel computation, the optimized trajectory $x^*_{t:t+N|t}$ with the minimum cost $J_s(x_t)$ discussed in (6) will be selected from among the $n+1$ groups of optimization problems. It will be tracked by the MPC controller introduced in III-D.

D. Overtaking Controller

After introducing the algorithm for trajectory generation, a low-level tracking controller with model predictive control used for overtaking will be discussed in this part. The constrained optimization problem is described as follows:

$$\underset{\tilde{u}_{t:t+N_c-1|t}, \omega_{1:N_c-1}}{\operatorname{argmin}} \sum_{k=0}^{N_c-1} \tilde{q}(\tilde{x}_{t+k|t}, \tilde{u}_{t+k|t}) + p_\omega(1-\omega_k)^2 \tag{11a}$$

$$\begin{aligned}
\text{s.t. } \tilde{x}_{t+k+1|t} =& A_{t+k|t}\tilde{x}_{t+k|t} + B_{t+k|t}\tilde{u}_{t+k|t} && \text{(11b)} \\
& + C_{t+k|t}, \ k=0,...,N_c-1, \\
\tilde{x}_{t+k+1|t} \in& \mathcal{X}, \tilde{u}_{t+k|t} \in \mathcal{U}, \ k=0,...,N_c-1, && \text{(11c)} \\
\tilde{x}_{t|t} =& \tilde{x}_t, && \text{(11d)} \\
h(\tilde{x}_{t+k+1|t}) \geq& \gamma\omega_k h(\tilde{x}_{t+k|t}), \ k=0,...,N_c-1, && \text{(11e)}
\end{aligned}$$

where (11b), (11c), (11d) describe the constraints for system dynamics (4), input/state bounds and initial conditions, respectively. The term $\tilde{q}(\tilde{x}_{t+k|t}, \tilde{u}_{t+k|t}) = ||\tilde{x}_{t+k|t} - x^*_{t+k|t}||^2_{\tilde{Q}_1} + ||\tilde{u}_{t+k|t}||^2_{\tilde{Q}_2}$ represents the stage cost, which tracks the desired trajectory $x^*_{t:t+N|t}$ optimized by the trajectory planner from Sec. III-C. Equation (11e) with $0 \leq \gamma < 1$ represents the discrete-time control barrier function constraints [44] with relaxation ratio ω_k, which could guarantee the system's safety by guaranteeing $h(\tilde{x}_{t+k|t}) > 0$ along the horizon with forward invariance. In this project, $h(\tilde{x}_{t+k|t}) = (\tilde{s}_{c,i} - \tilde{s}_c)^2 + (\tilde{e}_{y,i} - \tilde{e}_y)^2 - l^2 - d^2$ is used to represent the distance between the ego vehicle and other vehicles. The optimization (11) allows us to find the optimal control $u^*_t = \tilde{u}_{t|t}$ in a manner similar to MPC.

IV. Results

Having illustrated our autonomous racing strategy in the previous section, we now show the performance of proposed

Speed Range [m/s]	0 - 0.4	0.4 - 0.8	0.8 - 1.2	1.2 - 1.6
mean [s]	1.613	2.312	3.857	13.095
min [s]	0.8	1.2	1.8	3.5
max [s]	3.6	5.2	21.6	36.1

TABLE II: Time taken to overtake the leading vehicle travelling at different speeds. For each group of speed range of the leading vehicles, 100 cases were simulated. The ego vehicle starts from the track's origin. One other vehicle starts from a random position in the range of $10m \leq s_{c,i} \leq 30m$. The mean, min and max values show the average overtaking time, minimum overtaking time and maximum overtaking time for the corresponding group. In general, it takes more time to overtake faster moving vehicles on this track since they spend lesser time on the straight segments.

algorithm. The setup and results of numerical simulations will be presented in the following part.

A. Simulation Setup

In all simulations, all vehicles are 1:10 scale RC cars with a length of 0.4m and a width of 0.2m. Other vehicles drive along trajectories with fixed lateral deviation. The track's width is set to 2 m. The horizon lengths for trajectory planner and controller are $N_p = 12$, $N_c = 10$ and shared discretization time $\Delta t = 0.1s$. In our custom-designed simulator, both state and input noises are considered.

The optimization problems (7) and (11) are implemented in Python with CasADi [45] used as modeling language, and are solved with IPOPT [46] on Ubuntu 18.04 on a laptop with a CPU i7-9850 processor at a 2.6Ghz clock rate.

B. Racing With Other Vehicles

Snapshots shown in Fig. 1 illustrate example of overtaking behavior when the ego vehicle competes with three surrounding vehicles. The animations of more challenging overtaking behavior can be found on `https://youtu.be/1zTXfzdQ8w4`. As shown in TABLE I, the proposed racing planner could update at 25 Hz and could help the ego vehicle overtake multiple moving vehicles. By switching to a trajectory planning approach based on learning-based MPC, the ego vehicle is able to reach its speed and steering limit when there are no surrounding vehicles.

To better analyze the performance and limitations of our autonomous racing strategy in different scenarios, random tests are introduced under two groups. The first group of simulation aims to show the overtaking time for passing one leading vehicle with different speeds, and statistical results are summarized in TABLE II. We can observe that when the surrounding vehicles' speed reaches between 1.2m/s and 1.6m/s, much more time is needed for the ego vehicle to overtake the leading vehicle. This is because as the leading vehicle's speed increases, less space becomes available for the ego vehicle to drive safely. Especially in a curve, the ego vehicle's speed limit decreases when less space can be used to make a turn. Since more than half of our track is with curves, the ego vehicle needs to wait for a straight segment to accelerate to pass the leading vehicle.

The second group of simulation shows the proposed racing strategy's success rate to overtake multiple leading vehicles in one lap, and statistical results are summarized in TABLE III. We can find that when more than one surrounding vehicle exists, much more space would be occupied by other vehicles. As a result, the ego vehicle might not have enough space to accelerate to high speed to pass surrounding vehicles. Although in these cases, the ego vehicle can not overtake all surrounding vehicles after one lap, our proposed racing strategy can still guarantee the ego vehicle's safety along the track.

Speed Range [m/s]	0 - 0.4	0.4 - 0.8	0.8 - 1.2	1.2 - 1.6
Single	100 %	100 %	96 %	84 %
Two	100 %	100 %	98 %	66 %
Three	100 %	98 %	84 %	36 %

TABLE III: Overtaking success rate for the ego vehicle after one lap. For each group of speed range of the leading vehicles, 100 cases were simulated. The ego vehicle starts from the track's origin. Other vehicles start from a random position in the range of $5m \leq s_{c,i} \leq 15m$. One to three leading cars were simulated.

During our simulation, the mean solver time for our planner for single, two or three surrounding vehicles is 39.21ms, 39.41ms and 40.23ms. We also notice that when the number of surrounding vehicles is larger than three, the steady complexity still holds but the track becomes too crowded for the ego vehicle to achieve a high success rate of the overtaking maneuver. This validates the steady low computational complexity of our proposed planning strategy thanks to the parallel computation for multiple trajectory optimizations.

C. Racing Without Other Vehicles

As discussed in Sec. III-A, when there are no other surrounding vehicles, the ego vehicle adopts the learning-based MPC formulation for trajectory generation and control. In this paper, the learning-based MPC uses historical data from two previous laps and the initial data is calculated offline before the racing task. For the same setup as shown in Fig. 1, the ego vehicle is simulated to race without other agents. It's found that the ego vehicle slows down when there are surrounding vehicles. This is because the overtake maneuver happens in a hairpin curve and the curve's apex is occupied by other moving vehicles, resulting in less space being available for the ego vehicle and thus causing it to slow down to avoid a potential collision. After it passes all surrounding vehicles, the ego vehicle goes back to drive at its speed and steering limit to achieve time-optimal behavior.

V. Conclusion

In this paper, we have presented an autonomous racing strategy that enables an ego vehicle to enhance its lap timing performance while overtaking other moving vehicles. We have verified the performance of our proposed algorithm through numerical simulation, where several surrounding vehicles are simulated to start from random positions with random speeds on a track. Moreover, interaction between the ego vehicle and other surrounding vehicles will be considered in the future work. For instance, autonomous racing strategies such as blocking cars from overtaking are envisaged for the future.

REFERENCES

[1] M. Schmidt, C. Manna, J. H. Braun, C. Wissing, M. Mohamed, and T. Bertram, "An interaction-aware lane change behavior planner for automated vehicles on highways based on polygon clipping," *Robotics and Automation Letters*, vol. 4, no. 2, pp. 1876–1883, 2019.

[2] W. Yao, H. Zhao, F. Davoine, and H. Zha, "Learning lane change trajectories from on-road driving data," in *2012 IEEE Intelligent Vehicles Symposium*. IEEE, 2012, pp. 885–890.

[3] J. L. Vázquez, M. Brühlmeier, A. Liniger, A. Rupenyan, and J. Lygeros, "Optimization-based hierarchical motion planning for autonomous racing," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 2397–2403.

[4] S. He, J. Zeng, B. Zhang, and K. Sreenath, "Rule-based safety-critical control design using control barrier functions with application to autonomous lane change," in *2021 American Control Conference (ACC)*. IEEE, 2021, pp. 178–185.

[5] Y. Lyu, W. Luo, and J. M. Dolan, "Probabilistic safety-assured adaptive merging control for autonomous vehicles," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 10 764–10 770.

[6] H. Krasowski, X. Wang, and M. Althoff, "Safe reinforcement learning for autonomous lane changing using set-based prediction," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2020, pp. 1–7.

[7] D. Loiacono, L. Cardamone, and P. L. Lanzi, "Simulated car racing championship: Competition software manual," *arXiv preprint arXiv:1304.1672*, 2013.

[8] K. L. Talvala, K. Kritayakirana, and J. C. Gerdes, "Pushing the limits: From lanekeeping to autonomous racing," *Annual Reviews in Control*, vol. 35, no. 1, pp. 137–148, 2011.

[9] J. Y. Goh, T. Goel, and J. Christian Gerdes, "Toward automated vehicle control beyond the stability limits: drifting along a general path," *Journal of Dynamic Systems, Measurement, and Control*, vol. 142, no. 2, p. 021004, 2020.

[10] L. Hewing, A. Liniger, and M. N. Zeilinger, "Cautious nmpc with gaussian process dynamics for autonomous miniature race cars," in *2018 European Control Conference (ECC)*. IEEE, 2018, pp. 1341–1348.

[11] F. Fuchs, Y. Song, E. Kaufmann, D. Scaramuzza, and P. Dürr, "Super-human performance in gran turismo sport using deep reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4257–4264, 2021.

[12] T. Stahl, A. Wischnewski, J. Betz, and M. Lienkamp, "Multilayer graph-based trajectory planning for race vehicles in dynamic scenarios," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 3149–3154.

[13] A. Liniger and J. Lygeros, "A noncooperative game approach to autonomous racing," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 3, pp. 884–897, 2019.

[14] M. Wang, Z. Wang, J. Talbot, J. C. Gerdes, and M. Schwager, "Game-theoretic planning for self-driving cars in multivehicle competitive scenarios," *IEEE Transactions on Robotics*, 2021.

[15] A. Heilmeier, A. Wischnewski, L. Hermansdorfer, J. Betz, M. Lienkamp, and B. Lohmann, "Minimum curvature trajectory planning and control for an autonomous race car," *Vehicle System Dynamics*, 2019.

[16] U. Rosolia and F. Borrelli, "Learning model predictive control for iterative tasks. a data-driven control framework," *IEEE Transactions on Automatic Control*, vol. 63, no. 7, pp. 1883–1896, 2017.

[17] J. Kabzan, L. Hewing, A. Liniger, and M. N. Zeilinger, "Learning-based model predictive control for autonomous racing," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3363–3370, 2019.

[18] N. R. Kapania and J. C. Gerdes, "Learning at the racetrack: Data-driven methods to improve racing performance over multiple laps," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 8, pp. 8232–8242, 2020.

[19] H. Febbo, J. Liu, P. Jayakumar, J. L. Stein, and T. Ersal, "Moving obstacle avoidance for large, high-speed autonomous ground vehicles," in *2017 American Control Conference (ACC)*. IEEE, 2017, pp. 5568–5573.

[20] T. Brüdigam, A. Capone, S. Hirche, D. Wollherr, and M. Leibold, "Gaussian process-based stochastic model predictive control for overtaking in autonomous racing," in *2021 IEEE International Conference on Robotics and Automation (ICRA) Workshop for "Opportunities and Challenges with Autonomous Racing"*, 2021.

[21] J. Zeng, B. Zhang, and K. Sreenath, "Safety-critical model predictive control with discrete-time control barrier function," in *2021 American Control Conference (ACC)*. IEEE, 2021, pp. 3882–3889.

[22] A. Liniger, A. Domahidi, and M. Morari, "Optimization-based autonomous racing of 1: 43 scale rc cars," *Optimal Control Applications and Methods*, vol. 36, no. 5, pp. 628–647, 2015.

[23] N. R. Kapania, J. Subosits, and J. Christian Gerdes, "A sequential two-step algorithm for fast generation of vehicle racing trajectories," *Journal of Dynamic Systems, Measurement, and Control*, vol. 138, no. 9, 2016.

[24] S. A. Bonab and A. Emadi, "Optimization-based path planning for an autonomous vehicle in a racing track," in *IECON 2019-45th Annual Conference of the IEEE Industrial Electronics Society*, vol. 1. IEEE, 2019, pp. 3823–3828.

[25] D. Caporale, A. Settimi, F. Massa, F. Amerotti, A. Corti, A. Fagiolini, M. Guiggiani, A. Bicchi, and L. Pallottino, "Towards the design of robotic drivers for full-scale self-driving racing cars," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 5643–5649.

[26] S. Srinivasan, S. N. Giles, and A. Liniger, "A holistic motion planning and control solution to challenge a professional racecar driver," *IEEE Robotics and Automation Letters*, 2021.

[27] E. Alcalá, V. Puig, J. Quevedo, and U. Rosolia, "Autonomous racing using linear parameter varying-model predictive control (lpv-mpc)," *Control Engineering Practice*, vol. 95, p. 104270, 2020.

[28] A. Jain and M. Morari, "Computing the racing line using bayesian optimization," in *2020 59th IEEE Conference on Decision and Control (CDC)*. IEEE, 2020, pp. 6192–6197.

[29] C. Jung, S. Lee, H. Seong, A. Finazzi, and D. H. Shim, "Game-theoretic model predictive control with data-driven identification of vehicle model for head-to-head autonomous racing," in *2021 IEEE International Conference on Robotics and Automation (ICRA) Workshop for "Opportunities and Challenges with Autonomous Racing"*, 2021.

[30] J. Bhargav, J. Betz, H. Zheng, and R. Mangharam, "Track based offline policy learning for overtaking maneuvers with autonomous racecars," in *2021 IEEE International Conference on Robotics and Automation (ICRA) Workshop for "Opportunities and Challenges with Autonomous Racing"*, 2021.

[31] R. Wang, Y. Han, and U. Vaidya, "Deep koopman data-driven optimal control framework for autonomous racing," EasyChair, Tech. Rep., 2021.

[32] R. Oliveira, F. H. Rocha, L. Ott, V. Guizilini, F. Ramos, and V. Grassi, "Learning to race through coordinate descent bayesian optimisation," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 6431–6438.

[33] E. Perot, M. Jaritz, M. Toromanoff, and R. De Charette, "End-to-end driving in a realistic racing game with deep reinforcement learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 3–4.

[34] S. N. Wadekar, B. J. Schwartz, S. S. Kannan, M. Mar, R. K. Manna, V. Chellapandi, D. J. Gonzalez, and A. E. Gamal, "Towards end-to-end deep learning for autonomous racing: On data collection and a unified architecture for steering and throttle prediction," *arXiv preprint arXiv:2105.01799*, 2021.

[35] A. Remonda, S. Krebs, E. Veas, G. Luzhnica, and R. Kern, "Formula rl: Deep reinforcement learning for autonomous racing using telemetry data," in *IJCAI Workshop on Scaling-Up Reinforcement Learning: SURL@IJCAI*, 2019.

[36] Y. Song, H. Lin, E. Kaufmann, P. Dürr, and D. Scaramuzza, "Autonomous overtaking in gran turismo sport using curriculum reinforcement learning," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 9403–9409.

[37] U. Rosolia, S. De Bruyne, and A. G. Alleyne, "Autonomous vehicle control: A nonconvex approach for obstacle avoidance," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 2, pp. 469–484, 2016.

[38] N. Li, E. Goubault, L. Pautet, and S. Putot, "Autonomous racecar control in head-to-head competition using mixed-integer quadratic programming," Institut Polytechnique de Paris, Tech. Rep., 2021.

[39] R. Rajamani, *Vehicle dynamics and control*. Springer Science & Business Media, 2011.

[40] S. He, J. Zeng, and K. Sreenath, "Competitive car racing with multiple vehicles using a parallelized optimization with safety guarantee," *arXiv preprint arXiv:2112.06435*, 2021.

[41] L. Han, H. Yashiro, H. T. N. Nejad, Q. H. Do, and S. Mita, "Bezier curve based path planning for autonomous vehicle in urban environment," in *2010 IEEE intelligent vehicles symposium*. IEEE, 2010, pp. 1036–1042.

[42] J. Chen, P. Zhao, T. Mei, and H. Liang, "Lane change path planning based on piecewise bezier curve for autonomous vehicle," in *Proceedings of 2013 IEEE International Conference on Vehicular Electronics and Safety*. IEEE, 2013, pp. 17–22.

[43] X. Qian, I. Navarro, A. de La Fortelle, and F. Moutarde, "Motion planning for urban autonomous driving using bézier curves and mpc," in *2016 IEEE 19th international conference on intelligent transportation systems (ITSC)*. Ieee, 2016, pp. 826–833.

[44] J. Zeng, Z. Li, and K. Sreenath, "Enhancing feasibility and safety of nonlinear model predictive control with discrete-time control barrier functions," in *2021 Conference on Decision and Control (CDC)*, 2021, pp. 6137–6144.

[45] J. A. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "Casadi: a software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.

[46] L. T. Biegler and V. M. Zavala, "Large-scale nonlinear programming using ipopt: An integrating framework for enterprise-wide dynamic optimization," *Computers & Chemical Engineering*, vol. 33, no. 3, pp. 575–582, 2009.