Autonomous racecar control in head-to-head competition using Mixed-Integer Quadratic Programming

Nan Li[1] Eric Goubault[2] Laurent Pautet[1] Sylvie Putot[2]

***Abstract*— This work deals with the control of an autonomous racecar that should perform the fastest lap time on a track, while in presence of an opponent vehicle. Controlling the vehicle at its physical limit while ensuring collision-freeness is a challenging problem.**

We propose a Nonlinear Model Predictive Control (NMPC) model under a minimum time objective, which integrates the opponent vehicle's trajectory as a collision-avoidance constraint. By using a curvilinear coordinates system, progress time can be set as a direct optimization objective. The approximation of vehicle's shape is proposed and collision-avoidance constraints can therefore be represented efficiently. A safe control strategy is finally generated by a method based on Mixed-Integer Quadratic Programming (MIQP). We perform several experiments on our prototype implementation and discuss its performance issues.

I. Introduction

Autonomous racecar competition is an emerging topic that involves the control of vehicles at their physical limits, as well as highly complex collision avoidance methods. Some techniques developed in this context can also be transferred to non-racing autonomous cars. We consider here the problem of achieving the best lap time on a racing track, both without any opponent or given some (just one, for the sake of simplicity, in this paper) other competing vehicles. In our model, it is the ego vehicle's (EV) responsibility to avoid collision with an opponent or leading vehicle (LV) that it intends to overtake.

Several methods exist to compute control strategies for the problem of achieving the best lap time in absence of dynamic opponents. They can be classified in two approaches: either build a two-level controller [1] (a path planner at the higher level and a path follower at the lower level), or integrate the system dynamics, path constraints and control limits into a single Nonlinear Model Predictive Control (NMPC) problem [2], [3].

The overtaking problem has been considered in the context of highway driving, or for simple racing track configurations. In particular, Mixed-Integer Programming (MIP) is used in [4] to encode and solve the two-lane expressway overtaking problem. In [5], the vehicle's objective is to globally follow the center of a straight line track, with static obstacles avoidance.

Few works take into consideration the dynamic opponents in the autonomous racecar problem. In [6], trajectory planning is considered as an optimization problem by maximizing racecar's progress while penalizing deviation from a reference trajectory in a receding horizon way. A dynamic programming based high level corridor planner is proposed to generate convex constraints for obstacles. In [7], a similar idea of maximizing the progress of the racecar is used. The authors take into consideration the relative position and velocity between ego vehicle and opponent vehicle for setting up an extra constraint to avoid a potential collision. In [8], an off-line viability kernel computation is used to ensure feasibility, which allows to improve the performance of the online controller. The authors also take into account the interaction between two racecars formulated as a bimatrix game. In above articles, effort is made to maximize racecar's progress for indirectly optimizing its lap time performance.

Contributions.

In this work, we explore an alternative approach to achieve the fastest lap time in head-to-head racecar competition by extending single-racing mode NMPC-controller in [2]. This approach directly minimizes the progress time as the objective function while including **collision-freeness as constraint**. We introduce **an over-approximation** of the vehicle's shape in a curvilinear coordinates system. We propose a linear interpolation approach to **integrate the opponent vehicle's trajectory**, which can be computed online, as an avoidance constraint in the ego vehicle's trajectory controller. **Mixed-Integer Quadratic Programming** (MIQP) is used to encode and compute, at each step inside the prediction horizon, the optimal choice among all possible future overtaking strategies relying on the relative position of the 2 cars (ego vehicle situated at left, right, ahead or behind of the opponent). We test the feasibility of different overtaking scenarios with our prototype implementation. The influence of the length of prediction horizon is also discussed, with the objective to determine a satisfying trade-off: the prediction horizon needs to be sufficiently large for the problem to be always feasible, however larger horizon means larger optimization time, which may not be compatible with real time racing.

Organization.

The article is structured as follows. Section II introduces the track model and vehicle dynamics. Section III presents the formalization of single-vehicle racing problem and of the racecar head-to-head competition problem. The racing strategy and corresponding algorithms are described in Section IV. Section V presents numerical results for the proposed racing strategies.

[1]LTCI, Télécom Paris, Institut Polytechnique de Paris, France {nan.li,laurent.pautet}@telecom-paris.fr

[2]LIX, CNRS, Ecole Polytechnique, Institut Polytechnique de Paris, France {goubault,putot}@lix.polytechnique.fr

II. System Model

In this section, we introduce a curvilinear coordinate system (sub-section II-A) which is necessary for making progress time t as a dependent variable (sub-section II-B) and for finally making t as a direct optimization objective (next section). There are also several other advantages of using a curvilinear system: we can encode progress in a simple way; we can enforce the controlled trajectories to stay on track and to not collide with another vehicle using simple interval constraints.

A. Coordinate System Transformation

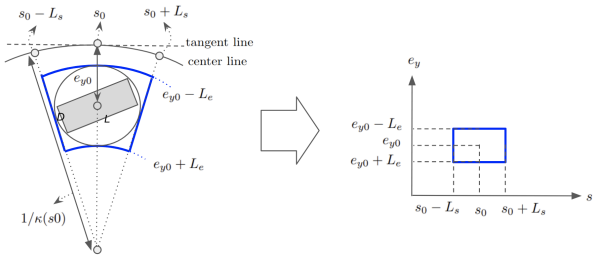

Fig. 1. Vehicles' shape is approximated as a set (blue sector) in the curvilinear coordinates system

As shown in Fig. 1, we adopt a curvilinear coordinate system [5], in which we represent the relative position of the vehicle's center of gravity with respect to its projection on the track's center-line as e_{y0}. The projection point is parametrized by the arc-length s_0 along the center-line.

We now define an over-approximation of the vehicle's occupied area in this coordinate system. Its center position is (s_0, e_{y0}). The vehicle's length and width are L and D, respectively. We firstly approximate vehicle's shape as a circle then project it into curvilinear coordinate. The vehicle's occupied area is

$$(s, e_y) \in [s_0 - L_s, s_0 + L_s] \times [e_{y0} - L_e, e_{y0} + L_e]$$

with

$$L_s = \frac{1}{\kappa(s)} \arcsin \frac{X}{\frac{1}{\kappa} - e_{y0}}, L_e = X, X = \frac{\sqrt{L^2 + D^2}}{2}$$

where $\kappa(s)$ is the curvature of local projection point.

B. Vehicle Dynamics

In this sub-section, we briefly introduce a bicycle dynamic model extended with a Pacejka lateral tire forces model. For details, we refer to the work [2], [3] and [6].

In the context of racing, one of main objectives is minimizing the progress time along the track. Using this curvilinear coordinate, we can transfer time-dependent system dynamics to (track) space-dependent dynamics. We can then explicitly set the time t as the objective function in the optimization algorithm that will identify the best trajectory. We define the vehicle's state (represented in Fig. 1) by $\xi = [e_y, e_\varphi, v_x, v_y, \omega, t, s, d, \delta]$, where e_y, e_ψ are the relative position of its center of gravity and relative orientation, v_x, v_y are the longitudinal and lateral velocities, ω is the change rate (angular velocities) of vehicle yaw ψ, t is the progress time, s is the progress length along the center-line of the track, d is the parameter for the motor engine, δ is the vehicle steering angle. The dynamics of the state vector ξ is

$$\frac{d}{dt}\xi = \frac{d\xi}{ds}\frac{ds}{dt} = \frac{d\xi}{ds}\dot{s},$$

where $\dot{s}$ is defined by the kinematic relation [3]:

$$\dot{s} = \frac{v_x cos(e_\psi) - v_y sin(e_\psi)}{1 - e_y \cdot \kappa(s)}.$$

Following [2], we define the vehicle dynamics for its center of gravity, with respect to the arc-length s along the center-line, as follows:

$$\frac{d}{ds}\begin{bmatrix} e_y \\ e_\psi \\ v_x \\ v_y \\ \omega \\ t \\ s \\ d \\ \delta \end{bmatrix} = \frac{1}{\dot{s}} \begin{bmatrix} v_x \sin(e_\psi) + v_y \cos(e_\psi) \\ \omega - \dot{s} \cdot \kappa(s) \\ \frac{1}{m}(F_{R,x} - F_{F,y} \sin\delta + m v_y \omega) \\ \frac{1}{m}(F_{R,y} + F_{F,y} \cos\delta - m v_x \omega) \\ \frac{1}{I_z}(l_f F_{F,y} \cos\delta - l_r F_{R,y}) \\ 1 \\ \dot{s} \\ \Delta d \\ \Delta\delta \end{bmatrix}, \quad (1)$$

where Δd and $\Delta\delta$ are control inputs, i.e. change rates for control signals d and δ; m is the vehicle weight; l_f and l_r are the distance between the vehicle center of mass and front / rear wheels respectively. The tire's longitudinal and lateral force is represented as following:

$$\begin{aligned} F_{R,x} &= (C_{m1} - C_{m2} v_x) \cdot d - C_{r0} - C_{r2} \cdot v_x^2 \\ F_{R,y} &= D_r \sin(C_r \arctan(B_r \alpha_R)) \\ F_{F,y} &= D_f \sin(C_f \arctan(B_f \alpha_F)) \\ \alpha_R &= \arctan((\omega l_r - v_y)/v_x) \\ \alpha_F &= -\arctan((\omega l_f + v_y)/v_x) + \delta \end{aligned} \quad (2)$$

$C_{m1,m2,r0,r2}, C_{r,f}, D_{r,f}$ are empirical parameters of the Pacejka tire model, for which we take the model identification value from [6] for the usage in the simulation part.

III. Problem Formalization

In this section, we will firstly present the problem formalization for the single vehicle racing problem. The lap time result of single vehicle racing is a baseline for testing the vehicle's performance in multiple vehicle racing. By adding constraints expressing collision-freeness, we then form a two vehicles head-to-head time-optimal racing problem.

A. Single Vehicle Racing

Finding a control minimizing the lap time is naturally expressed as an Optimal Control Problem (OCP). Piecewise constant control parameterization changes a continuous OCP into a Model Predictive Control (MPC) problem, which can be solved efficiently. We use a multiple shooting method for an horizon of N control-steps. The resulting sets of

constraints can then be solved by Non-Linear Programming (NLP) optimisation. Following the proposal of [2], we solve this MPC problem by sequentially solving Quadratic Programs (QP) based on an exact Hessian matrix expansion.

For the single-vehicle racing problem, the objective is to minimize the progress time while keeping the car within the track boundaries. The corresponding NLP-form formalization is:

$$\begin{aligned} \min_{u_i(s)} \quad & t_N \\ s.t. \quad & \xi'_{i+1} = f_{dyn}(\xi_i, u_i),\ i = 0, ..., N \\ & \xi_i \in [\underline{\xi}, \overline{\xi}],\ i = 0, ..., N+1 \\ & u_i \in [\underline{u}, \overline{u}],\ i = 0, ..., N, \end{aligned} \tag{3}$$

where ξ_i is the state vector $[e_y, e_\varphi, v_x, v_y, \omega, t, s, d, \delta]$ and u_i is the control vector $[\Delta d, \Delta\delta]$.

We add to (3) the following extra constraint which forces the vehicle to stay within the track:

$$-L_t + X \leq e_y \leq +L_t - X, X = \frac{\sqrt{L^2 + D^2}}{2} \tag{4}$$

where L_t is the half of track length.

B. Two Vehicles Head-to-Head Competition

In two vehicles head-to-head competition, we consider the scenario in which there exists a Leading Vehicle (LV) ahead of Ego Vehicle (EV). When LV is closely ahead of EV, we should make sure that the control strategy of EV will avoid collision with LV. The collision-avoidance constraint for EV is that there should be no intersection between their (approximated) occupied areas:

$$\begin{aligned} & [s_0^{EV} - L_s^{EV}, s_0^{EV} + L_s^{EV}] \\ & \times [e_{y0}^{EV} - L_e^{EV}, e_{y0}^{EV} + L_e^{EV}] \\ \cap & [s_0^{LV} - L_s^{LV}, s_0^{LV} + L_s^{LV}] \\ & \times [e_{y0}^{LV} - L_e^{LV}, e_{y0}^{LV} + L_e^{LV}] = \emptyset \end{aligned} \tag{5}$$

We add this extra condition into the single-vehicle racing formulation (3) to obtain the formalization for the two vehicles competition problem. In the next section we reformulate this condition into a mixed-integer form.

IV. Racing Strategy and Algorithms

From now on, we consider head-to-head racing. We make the assumption that when EV is behind LV, it is EV's responsibility to avoid collision. We suppose that EV knows about the opponent's state information by inference or communication. Moreover, we assume that LV's trajectory is calculated by the time-optimal control same with EV but in the absence of competitors, which is also based on NMPC model. Indeed, one difficulty comes from that the N different steps of EV or LV in prediction horizon are discrete N states based on progress s. For EV at given progress s, we need to interpolate LV's position as a function of time t. We introduce a linear interpolation method in Section IV-A. LV's positions are then integrated to establish the collision avoidance constraints. Combining this extra constraint with the original NLP formulation (3), we formulate a Mixed-Integer Quadratic Programming (MIQP) problem to generate a collision-free time-optimal trajectory for EV in Section IV-B.

A. Linear interpolation of LV's states

EV's trajectory is discretized at progress points $i = 1, \ldots, M$. At each of these progress points, EV has a prediction horizon of length N. The progress time t_N^{EV} is the objective to be minimized and the time series t_j^{EV}, $j = 1, \ldots, N$ is needed for inferring LV's N positions.

According to the planning information of EV at last progress points $i-1$, we initialize a first guess for EV's N-steps progress time series. Using formulation (3) and (4), we can calculate LV's (discrete) trajectory. Combining with t_j^{EV}, we infer LV's position between its sampling points, say between $[s_1^{LV}, {e_y}_1^{LV}, t_1^{LV}]$ and $[s_2^{LV}, {e_y}_2^{LV}, t_2^{LV}]$. We build a linear interpolation of LV's position around the located points at step $j, j = 1, \ldots, N$:

$$\begin{cases} s_j^{LV}(t_j^{EV}) = \frac{s_2^{LV} - s_1^{LV}}{t_2^{LV} - t_1^{LV}} \cdot (t_j^{EV} - t_1^{LV}) + s_1^{LV} \\ {e_y}_j^{LV}(t_j^{EV}) = \frac{{e_y}_2^{LV} - {e_y}_1^{LV}}{t_2^{LV} - t_1^{LV}} \cdot (t_j^{EV} - t_1^{LV}) + {e_y}_1^{LV} \end{cases} \tag{6}$$

B. Formulation of MIQP problem

During the execution of our MIQP-based NMPC algorithm, we first linearize dynamics and constraints around the state and control values produced from last progress point $i-1$. LV's positions $(s^{LV}, e_y^{LV})_j, j = 1, \ldots, N$ are interpolated as explained in Section IV-A. After solving a single MIQP, we generate a new N-steps prediction. If the corresponding solution does not reach the required precision, we linearize again the dynamics and constraints around the new state and control values. We use the Karush–Kuhn–Tucker (KKT) condition as the precision indicator for indicating the optimality of the solution and the violation of constraints. If the KKT value does not reach the required precision after the predefined maximum iteration number, the solution taken from the last progress point $i-1$ is reused.

The key point here is to build the MIQP formulation form, i.e. to transfer the general definition of the collision-avoidance constraint (formulation (5)) into a mixed-integer form. For EV at progress point $i \in (1, \ldots, M)$, the collision-avoidance constraint for each step $j = 1, .., N$ in the prediction horizon is:

$$\begin{aligned} & (A)\ s_j^{LV} + (L_s)_j^{LV} \leq s_j^{EV} - (L_s)_j^{EV} \\ OR\ & (B)\ s_j^{EV} + (L_s)_j^{EV} \leq s_j^{LV} - (L_s)_j^{LV} \\ OR\ & (C)\ {e_y}_j^{LV} + (L_e)_j^{LV} \leq {e_y}_j^{EV} - (L_e)_j^{EV} \\ OR\ & (D)\ {e_y}_j^{EV} + (L_e)_j^{EV} \leq {e_y}_j^{LV} - (L_e)_j^{LV} \end{aligned} \tag{7}$$

We name each of these constraints using functions f_A, f_B, f_C and f_D so that these become: $(f_A(j) \leq 0) \vee (f_B(j) \leq 0) \vee (f_C(j) \leq 0) \vee (f_D(j) \leq 0)$.

The above constraints correspond to the following configurations: (A) EV is ahead of LV; (B) EV is behind LV; (C) EV is at the left of LV; (D) EV is at the right of LV. However

there is an overlap among these 4 configurations. We refine them into 4 new non-overlapping configurations, by adding to the cases (C) and (D) the condition that EV is neither totally ahead of LV nor totally behind of LV. We formally write them as:

$$\begin{aligned}&(f_A(j) \leq 0)\\ \vee&(f_B(j) \leq 0)\\ \vee&(f_C(j) \leq 0 \wedge (f_A(j) > 0 \wedge f_B(j) > 0))\\ \vee&(f_D(j) \leq 0 \wedge (f_A(j) > 0 \wedge f_B(j) > 0))\end{aligned}$$

We then use the big-M method [9] to encode this disjunction into a more standard set of constraints:

$$\begin{cases} f_A(j) \leq c_1 \cdot M \\ f_B(j) \leq c_2 \cdot M \\ f_C(j) \leq c_3 \cdot M \\ -f_A(j) \leq c_3 \cdot M \end{cases} \quad \begin{cases} -f_B(j) \leq c_3 \cdot M \\ f_D(j) \leq c_4 \cdot M \\ -f_A(j) \leq c_4 \cdot M \\ -f_B(j) \leq c_4 \cdot M \end{cases} \tag{8}$$

where $c_k \in \{0, 1\}, k = 1, 2, 3, 4$ are binary variables satisfying $c_1 + c_2 + c_3 + c_4 \leq 3$, and M is a sufficiently large positive number. We finally use the method from [10] to reduce the number of binary variables from 4 to 2:

$$\begin{cases} f_A(j) \leq a_1 \cdot M \\ f_B(j) \leq a_2 \cdot M \\ f_C(j) \leq a_3 \cdot M \\ -f_A(j) \leq a_3 \cdot M \end{cases} \quad \begin{cases} -f_B(j) \leq a_3 \cdot M \\ f_D(j) \leq a_4 \cdot M \\ -f_A(j) \leq a_4 \cdot M \\ -f_B(j) \leq a_4 \cdot M \end{cases} \tag{9}$$

where $a_1 = 1 + c_1 - c_2$, $a_2 = 1 - c_1 + c_2$, $a_3 = c_1 + c_2$, $a_4 = 2 - c_1 - c_2$, c_1 and c_2 are binary variables. If $c_1 = 0, c_2 = 1$, the first constraint is active and other constraints are relaxed. If $c_1 = 1, c_2 = 0$, the second constraint is active. If $c_1 = c_2 = 0$, the third group of constraint is active. If $c_1 = c_2 = 1$, the last group of constraint is active.

Combining (3), (4), (6) and the constraints (9) for all predicted step $j = 1, \ldots, N$, we formulate a MIQP problem which can be solved in sequence by a MIQP solver to generate a N-steps no-collision time-optimal control strategy for EV at progress point i.

V. Simulation Results

A. Implementation and experimental setup

Based on our problem formalization, we use the ACADO Code Generation Tool [11] to generate a framework for sequentially solving Hessian based QP problem [12]. An interface to QP solvers is provided inside this framework. We selected the qpOASES [13] as QP solver option for the single-vehicle mode. We also wrote a wrapper to call the GUROBI [14] solver for using MIQP method in head-to-head racing mode. The following experiments are performed on a standard laptop featuring an Intel i7 CPU and 32 GB of RAM under Ubuntu 18.04.

We test two classic tracks for racecars: track 1, used in [2] with full length of $8.7\ m$ and track 2, used in [6] with full length of $18.0\ m$. Both track widths are $0.34\ m$.

TABLE I

Physical constraint for state variables

Variable	Range
e_y	$[-0.17, +0.17]\ m$
e_ψ	$[-1.5, +1.5]\ rad$
v_x	$[0.05, +1.6]\ m/s$
v_y	$[-1.0, +1.0]\ m/s$
r	$[-8.0, +8.0]\ rad$

Variable	Range
d	$[-1.0, +1.0]$
δ	$[-0.6, +0.6]\ rad$
Δd	$[-10.0, +10.0]\ s^{-1}$
$\Delta\delta$	$[-10.0, +10.0]\ rad/s$

We use the vehicle model identification described in [6]. Used constraints are listed in Table I. We assume that LV has identical model parameters as EV but with a different limitation for the maximum longitudinal velocity - $v_x \in [0.05, 1.2]$ - for ensuring the possibility for EV to overtake LV. In the simulation, LV's trajectory is calculated using (3) and (4) off-line for simplicity. In a real-world setting, LV's trajectory can be calculated online.

We set the KKT value to 10^{-4}: the optimization process ends when the KKT condition is lower than this value, which means that it is optimal enough and satisfies well the constraints. The maximum number of iterations for the MIP method is 20. The step length in the prediction horizon is selected to be $0.06\ m$ which is close to the length of vehicles. The horizon length N has an important impact on the lap time performance and calculation time, this is discussed in the next sub-section.

For the simulation of single-vehicle racing, EV is initially located at the $(s, e_y) = (0, 0)$ and finishes a full lap.

For two-car head-to-head racing, LV is initially located at point $(0, 0)$. EV is located at 3 possible initial positions - $(0, 0)$, $(0, -0.1)$ and $(0, 0.1)$ - to express a certain diversity of competition scenarios. In some test scenarios, EV starts running after LV reaches progress length of $0.1\ m, 0.2\ m, \ldots, 0.8\ m$ for track 1 and $0.1\ m, 0.2\ m, \ldots, 1.5\ m$ for track 2. The vehicles' initial longitudinal speed is set to $1.0\ m/s$. This constitutes 24 and 45 scenarios respectively for track 1 and track 2. These cover overtaking at many different positions in these two track layouts.

B. Single-vehicle racing performance

TABLE II

Simulation result for the single-vehicle racing

	Horizon length N	Lap time [s]	Mean calculation time per step [s]
Track 1	15	4.852	0.137
	20	4.802	0.171
	30	4.773	0.245
	40	4.768	0.355
	50	4.767	0.483
Track 2	15	10.189	0.118
	20	10.107	0.140
	30	10.064	0.205
	40	10.059	0.307
	50	10.059	0.460

We test the single-vehicle racing scenario as a baseline to compare different lengths N of prediction horizon under

current configuration. The result is shown in Table II. For track 1, the lap times achieved for $N = 40$ and 50 indicate that the minimum lap time is around $4.767\ s$. This shows that $N = 30$ achieves a good enough lap time (0.13% slower than $N = 50$) while maintaining relatively low calculation time. $N = 15$ is also interesting because of its low computational cost. Experiments on track 2 show very similar results. We use these two different lengths of prediction horizon $N = 15$ and $N = 30$ to perform the experiments in the case of two-vehicles head-to-head competition.

C. Head-to-head racing performance

TABLE III
SIMULATION RESULT FOR TWO CAR HEAD-TO-HEAD RACING

	Horizon length N	# of cases where collision happens	Average lap time [s]	Average calculation time per step before overtaking [s]
Track 1	15	3/24	4.942	0.247
	30	0/24	4.899	0.905
Track 2	15	0/45	10.277	0.243
	30	0/45	10.148	0.832

Table III summarizes the lap time and computation time for all scenarios. As can be expected, a longer horizon yields better lap time and a higher computation cost.

The trade-off between trajectory optimality and computation time naturally depends on the available computing resources. The computational cost increases with the prediction horizon. On the other hand, a sufficient horizon is necessary for finding feasible trajectories. Three failed cases with $N = 15$ on track 1 are due to a too short time horizon: EV is too close to LV with a too high (longitudinal or angular) speed and it has not predicted this situation in advance because of a short horizon.

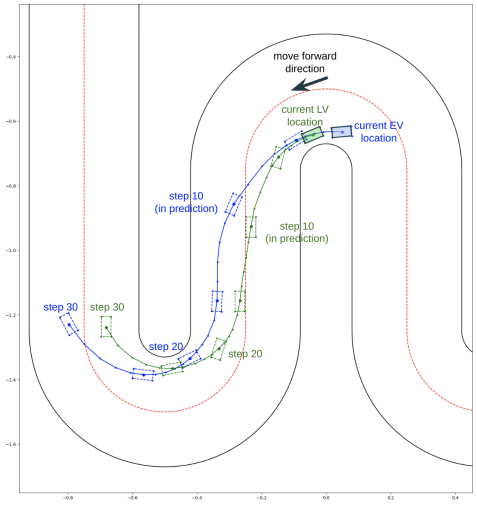

Fig. 2. A typical example of predicted trajectories of EV and LV. Rectangles: EV (blue) and LV (green) at actual location, followed by predicted positions at steps 5 / 10 / 15 / 20 / 25 / 30; dot lines: EV's and LV's predicted trajectories.

A typical example of calculation result for a given progress point is presented in Fig. 2. It corresponds to one of the 24 scenarios that we test on track 1: EV is initially located at $(0, 0)$ and starts running when LV reaches $0.8\ m$. The figure shows the moment that EV arrives at $s = 5.04\ m$.

From this example, we observed the following behavior in EV's prediction horizon: EV plans to follow LV from step 1 to 10 (condition B in (7) is active), to overtake LV at the right from step 11 to 19 (condition D in (7) is active), to be completely ahead of LV at step 20 (condition A in (7) is active) and to keep this advantage until step 26, to keep at the left of LV at the last 4 steps (condition C in (7) is active).

D. Calculation time issues

The previous sub-sections demonstrated the good behavior of the algorithm. However, with the current configuration, the average progress time per step is about $30\ ms$ which is lower than the current calculation time per step (for N=15, it is around $140\ ms$ on single mode and $250\ ms$ on head-to-head racing mode on average). It shows the difficulty of the implementation on a real-world racecar of the NMPC-based controller with the MIQP method encoding non-collision constraints.

There are several possibilities to solve this problem in the future: one possibility is to simplify the decision combinatorics using the observed fact that the overtaking strategy is often quite stable between successive steps (for example in Fig. 2, we showed successive following, successive right-overtaking and successive left-overtaking behavior); another possibility is to explore the problem structure of MIQP method and take the advantage of the multi-core system or the GPU (such as the GPU that equips the F1tenth racecar).

VI. CONCLUSION

Different from the previous effort in literature that forms an optimization problem to maximize the racecar's progress distance, we explore an alternative approach to achieve the best lap time which sets the progress time as a direct objective. We set up the method for approximating vehicle's shape in curvilinear coordinate system, integrating opponent's trajectory into optimization constraints, and modelling the collision-avoidance constraints using mixed-integer form. The simulation on our prototype implementation showed the effectiveness of the proposed algorithm.

In future work, we consider further focusing on the feasibility of these methods in a real-world setting, and in particular implementing them on a F1tenth racecar. This involves studying dynamic adaptation of prediction horizons, studying strategies that reduce the combinatorics of constraints, and taking advantage of the computation on multi-core platforms. Finally, in the longer term we consider combining such approach with online reachability analysis that monitors safety of decisions.

REFERENCES

[1] A. Gray, Y. Gao, T. Lin, J. K. Hedrick, H. E. Tseng, and F. Borrelli, "Predictive control for agile semi-autonomous ground vehicles using motion primitives," *2012 American Control Conference (ACC)*, pp. 4239–4244, 2012.

[2] R. Verschueren, M. Zanon, R. Quirynen, and M. Diehl, "Time-optimal race car driving using an online exact hessian based nonlinear mpc algorithm," *ECC 2016*, pp. 141–147.

[3] J. V. Frasch, A. Gray, M. Zanon, H. J. Ferreau, S. Sager, F. Borrelli, and M. Diehl, "An auto-generated nonlinear mpc algorithm for real-time obstacle avoidance of ground vehicles," *2013 European Control Conference*, pp. 4136–4141, 2013.

[4] F. Molinari, N. N. Anh, and L. del Re, "Efficient mixed integer programming for autonomous overtaking," *ACC 2017*.

[5] Y. Gao, A. Gray, J. V. Frasch, T. Lin, E. Tseng, J. K. Hedrick, and F. Borrelli, "Spatial predictive control for agile semi-autonomous ground vehicles," in *11th International Symposium on Advanced Vehicle Control*, no. 2, 2012, pp. 1–6.

[6] A. Liniger, A. Domahidi, and M. Morari, "Optimization-based autonomous racing of 1: 43 scale rc cars," *ArXiv abs/1711.07300*, 2017.

[7] M. Kloock, P. Scheffe, L. Botz, J. Maczijewski, B. Alrifaee, and S. Kowalewski, "Networked model predictive vehicle race control," *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*.

[8] A. Liniger and J. Lygeros, "A noncooperative game approach to autonomous racing," *IEEE Transactions on Control Systems Technology*, vol. 28, pp. 884–897, 2020.

[9] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[10] I. Prodan, S. Olaru, C. Stoica, and S.-I. Niculescu, "Collision avoidance and path following for multi-agent dynamical systems," in *ICCAS 2010*, pp. 1930–1935.

[11] B. Houska, H. J. Ferreau, and M. Diehl, "An auto-generated real-time iteration algorithm for nonlinear mpc in the microsecond range," *Autom.*, vol. 47, pp. 2279–2285, 2011.

[12] P. T. Boggs and J. W. Tolle, "Sequential quadratic programming," *Acta numerica*, vol. 4, no. 1, pp. 1–51, 1995.

[13] H. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock, and M. Diehl, "qpoases: A parametric active-set algorithm for quadratic programming," *Mathematical Programming Computation*, vol. 6, no. 4, 2014.

[14] "Inc. gurobi optimizer reference manual, 2015."