Game-Theoretic Model Predictive Control with Data-Driven Identification of Vehicle Model for Head-to-Head Autonomous Racing

Chanyoung Jung[1], Seungwook Lee[1], Hyunki Seong[1], Andrea Finazzi[2] and David Hyunchul Shim[1]

Abstract— Resolving edge-cases in autonomous driving, head-to-head autonomous racing is getting a lot of attention from the industry and academia. In this study, we propose a game-theoretic model predictive control (MPC) approach for head-to-head autonomous racing and data-driven model identification method. For the practical estimation of nonlinear model parameters, we adopted the hyperband algorithm, which is used for neural model training in machine learning. The proposed controller comprises three modules: 1) game-based opponents' trajectory predictor, 2) high-level race strategy planner, and 3) MPC-based low-level controller. The game-based predictor was designed to predict the future trajectories of competitors. Based on the prediction results, the high-level race strategy planner plans several behaviors to respond to various race circumstances. Finally, the MPC-based controller computes the optimal control commands to follow the trajectories. The proposed approach was validated under various racing circumstances in an official simulator of the Indy Autonomous Challenge. The experimental results show that the proposed method can effectively overtake competitors, while driving through the track as quickly as possible without collisions.

I. INTRODUCTION

Recently, prominent races related to autonomous driving have been actively organized to solve real-world problems in "edge case" scenarios beyond the autonomous driving technology in general situations, such as Roborace [1], Indy Autonomous Challenge [2], and DARPA-RACER [3]. Autonomous racing can be divided into three categories: 1) time trial, 2) 1:1 racing, and 3) head-to-head autonomous racing, and all three types of racing are driven by limits of handling; however, head-to-head autonomous racing, where there is more than one opponent, presents more technical difficulties. In this study, we focused on head-to-head autonomous racing.

The MPC framework, which finds the optimal control commands based on the system model, while satisfying a set of constraints, is the most widely used controller for high-speed autonomous driving. In [4], they formulated the objective function of MPC which aims to maximize the progress along the reference path by integrating a non-linear vehicle model for time trial racing. In [5], they proposed a model predictive contouring control (MPCC) approach that

[1]Chanyoung Jung, Seungwook Lee, Hyunki Seong, and David Hyunchul Shim are with the School of Electrical Engineering, Korea Advanced Institute of Science and Technology, Daejeon, South Korea. `cy.jung@kaist.ac.kr`, `seungwook1024@kaist.ac.kr`, `hynkis@kaist.ac.kr`, `hcshim@kaist.ac.kr`

[2]Andrea Finazzi is with the KAIST Robotics Program, Korea Advanced Institute of Science and Technology, Daejeon, South Korea. `finazzi@kaist.ac.kr`

(Chanyoung Jung, Seungwook Lee and Hyunki Seong contributed equally to this work)

can follow the reference path and avoid stationary obstacles. They demonstrated the real-time capability of the proposed method through real-world experiments using a 1:43 scale vehicle. However, most of the existing studies have focused mainly on the MPC formulation based on the assumption that the vehicle and tire model parameters are known in advance.

Unlike time trials, head-to-head racing players should be able to predict future trajectories that reflect the intentions and strategies of other opponents and drive through the track as quickly as possible, while balancing safety and aggressiveness. In [6], an MPC that can overtake the other competitors was proposed. The MPC objective function incorporates the euclidean distance between the ego-vehicle and the opponents, penalizing collision-prone trajectories. The authors of [7] proposed a game-theoretic planner for overtaking in a two-car racing scenario. They used an iterative best response algorithm (IBR) that seeks for the Nash equilibrium within the joint trajectory of the two vehicles. Similarly, in [8], a game-theoretic approach was proposed to model racing as a non-cooperative non-zero-sum game, while assuming open-loop information structures. Although various approaches have been suggested for autonomous racing with others, head-to-head racing remains challenging, and much work remains to be done.

Herein, we present the Stackelberg game-theoretic MPC for head-to-head autonomous racing, along with a data-driven optimization-based approach to identify the system model parameters that underlie MPC. Specifically, for model identification, we applied the hyperband algorithm, which is a widely used practical approach in neural model learning. With the obtained models, we designed a race-oriented controller, which can respond to various race circumstances without collision with surrounding vehicles. Our proposed controller comprises three modules: 1) Stackelberg game-based opponents' trajectory predictor, 2) high-level race strategy planner, and 3) MPC-based low-level controller. Finally, the proposed method was verified based on a head-to-head racing scenario in a realistic vehicle simulator.

II. MODEL IDENTIFICATION USING HYPERBAND ALGORITHM

For MPC-based vehicle control, highly nonlinear model parameters are required. The inaccuracy in the parameters can result in model error and it is propagated over the MPC horizon. Consequently, this reduces the prediction accuracy of MPC and lowers the optimality of the solution. Furthermore, finding the correct model parameters is challenging and requires many heuristic search processes.
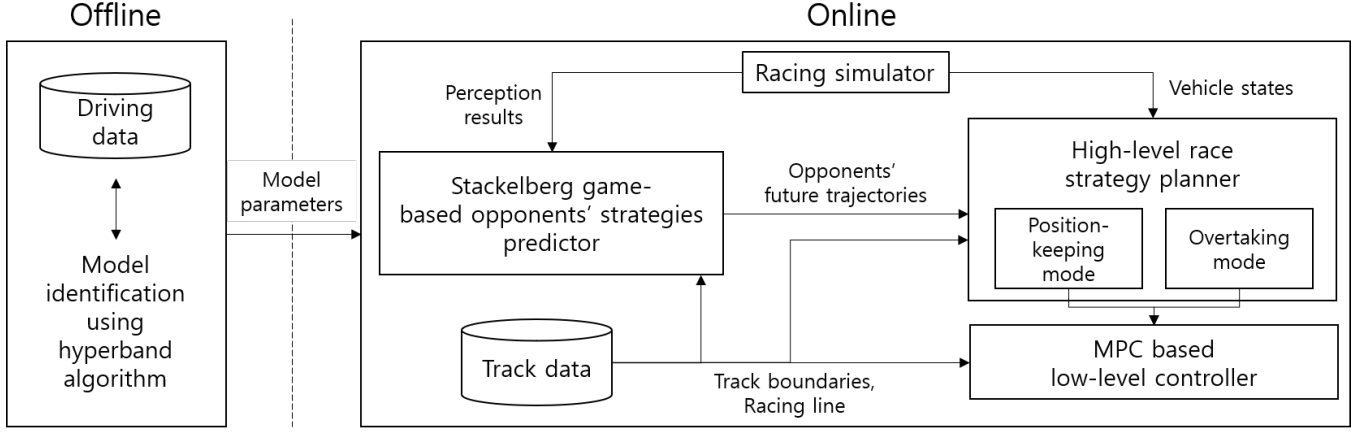
Fig. 1: Overview of the proposed approach for head-to-head autonomous racing.

To address this challenging but practical issue, we adopted a hyperparameter optimization scheme to find the system model parameters in a data-driven approach, which is widely used in the field of machine learning. Specifically, we used the hyperband [9] algorithm to obtain the parameters of a simplified Pacejka tire model [10] for vehicle dynamics. The hyperband algorithm is a variation of random search, but with some explore-exploit theory to find the best hyperparameters for each configuration based on an evaluation loss. It defines budget B, which is the total number of epochs required to find the hyperparameter configuration. B determines the number of samples and the number of iterations spent on each sampled configuration. Comparing the evaluation loss for each sampled configuration, hyperband excludes the configuration with a large loss and allocates more budget for the configuration with a low loss. It repeats this process until the optimal configuration is chosen. To handle the local minimum problem, we apply Gaussian mutation process [11]. New parameter configurations can be generated by injecting Gaussian noise to the selected configurations, and this increases the likelihood of finding a well-fitted solution with a lower evaluation loss.

Algorithm 1 summarizes the aforementioned parameter search process, where R is the maximum budget amount and η is an input that controls the proportion of the discarded configurations. For parameter optimization, we define the following three functions in the hyperband.

- *get_hyperparameter_configuration(n):* a function that returns a set of n hyperparameter configurations from normal distribution defined over the configuration space.
- *mutate_then_return_eval_loss(t, r_i):* a function that receives a hyperparameter configuration t and a resource (budget) allocation r as arguments and returns the evaluation loss after mutating the configuration for the allocated resources.
- *select_top_k_configuration$(T, L, \lfloor n_i/\eta \rfloor)$:* a function that receives a set of hyperparameter configurations T and their corresponding evaluation losses L and returns the top k performing configurations.

Algorithm 1 Hyperband algorithm for model identification parameter search

---

Input: R, η (default $\eta = 3$)

1:   $s_{max} \leftarrow \lfloor \log_\eta(R) \rfloor, B = (s_{max} + 1)R$
2:   **for** $s \in \{s_{max}, s_{max} - 1, ..., 0\}$ **do**
3:      $n = \lceil \frac{B}{R} \frac{\eta^s}{(s+1)} \rceil, r = R\eta^{-s}$
4:      $T = $ get_hyperparameter_configuration(n)
5:      **for** $i \in \{0, ..., s\}$ **do**
6:         $n_i = \lfloor n\eta^{-i} \rfloor$
7:         $r_i = r\eta^i$
8:         $L = \{$mutate_then_return_eval_loss$(t, r_i) : t \in T\}$
9:         $T = $ select_top_k_configuration$(T, L, \lfloor n_i/\eta \rfloor)$

Output: Configuration with the smallest loss L

III. GAME-THEORETIC MODEL PREDICTIVE CONTROLLER

A. Overall Structure

The proposed game-theoretic MPC is designed to infer the optimal control command to drive through the track as quickly as possible without collision with surrounding opponents in a head-to-head racing scenario. The proposed controller is composed of three modules: 1) game-based opponents' trajectory predictor, 2) high-level race strategy planner, and 3) MPC-based low-level controller. The predictor establishes a two-player game between surrounding vehicles based on the perception results and solves them sequentially to set the future trajectory of the surrounding vehicles. At this time, we assumed that all the vehicle dynamics are same, and the payoff, the main gradient of the game, was shared as an objective function of the MPC. Based on the predicted trajectories, the racing strategy planner selects the strategies according to the predefined race mode (position-keeping or overtaking), balancing between aggressiveness and conservativeness. Finally, the MPC-based low-level controller is responsible for generating the optimal control command along the race strategy. The overall structure of our approach is shown in Fig. 1.
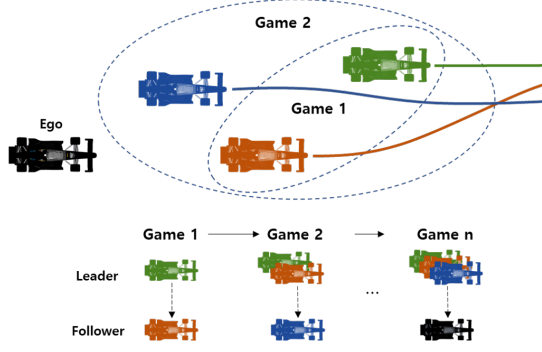
Fig. 2: Visualization of the structure of the proposed Stackelberg game implementation. A two-player game (leader vs. follower) is turned into an n-player game by recursively stacking the leader's trajectories.

B. Extend Stackelberg Game to N-Player Game

The Stackelberg game is a simple two-player sequential game played by a leader and a follower. In this game, the leader first commits his strategy, and the follower observes the leader's strategy and responds to it while both are trying to maximize their own payoff.

To extend the two-player game to an n-player game, we sequentially establish independent Stackelberg games between the surrounding vehicles. Starting from the leading one, we solve each game recursively, as shown in Fig. 2. In each game, we assumed that two vehicles are trying to maximize their own progress, whereas only the follower takes into account the collision. Therefore, each player's payoff function can be formulated as the objective function of MPC, as described in Section III-C. The output of each game (one trajectory for each player) is passed to consecutive games and set as additional leader strategies. To keep the computational load acceptable, we only considered the vehicles within a 100 m range from the ego-vehicle.

C. MPC-based Low-level Controller and High-level Race Strategy Planner

This section begins with the vehicle dynamics model used in this study, and the path-following MPC problem is formulated. Subsequently, it represents the detailed contents of the MPC-based low-level controller and high-level strategy planner for head-to-head autonomous racing.

1) Vehicle Model: The vehicle was modeled using a dynamic bicycle model [12]. For the sake of simplicity, the vehicle was modeled as a single rigid body expressed by mass m and inertia I_z. In the bicycle model, lateral tire forces $F_{F,y}$ and $F_{R,y}$ were applied at the front and rear wheels, respectively. Our race vehicle was a rear-wheel drive; therefore, the longitudinal tire force $F_{R,x}$ was applied only at the rear wheel. The pitch, roll, and vertical dynamics were ignored, and only the motion on a flat surface was considered. The resulting vehicle dynamics are shown in Fig. 3, and the equation of the dynamics is given in Equation 1.
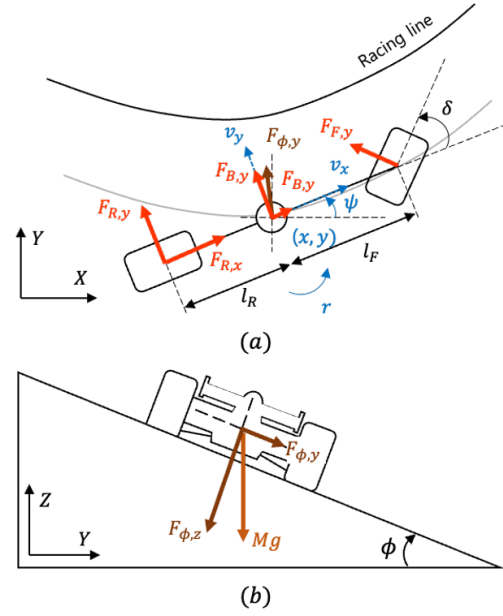

Fig. 3: Schematic of vehicle dynamics model (a) in the yaw and (b) roll planes.

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{\psi} \\ \dot{v}_x \\ \dot{v}_y \\ \dot{r} \end{bmatrix} = \begin{bmatrix} v_x \cos\psi - v_y \sin\psi \\ v_x \sin\psi + v_y \sin\psi \\ r \\ \frac{1}{m}(F_{R,x} - F_{F,y}\sin\delta + mv_yr + F_{B,x}) \\ \frac{1}{m}(F_{R,y} + F_{F,y}\cos\delta - mv_xr + F_{B,y}) \\ \frac{1}{I_z}(F_{F,y}l_F\cos\delta - F_{R,y}l_R) \end{bmatrix}, \quad (1)$$

where X, Y and ψ are the position and orientation in a global coordinate, respectively. v_x, v_y and r are the longitudinal, lateral velocities and yaw rate, respectively. l_F and l_R are the distances from the center of gravity to the front and rear wheels, respectively. The model is represented by the 6-dimensional state $\mathbf{x} = [X, Y, \psi, v_x, v_y, r]^T$ and 2-dimensional input $\mathbf{u} = [\delta, D]^T$, where δ and D are the steering angle and throttle command, respectively. Note that the braking command is included in the negative range of the D term.

The tire forces are represented by the simplified Pacejka tire model,

$$\alpha_F = \arctan\frac{v_y + l_F r}{v_x} - \delta,$$
$$\alpha_R = \arctan\frac{v_y - l_R r}{v_x}, \quad (2)$$
$$F_{i,y} = D_i \sin C_i \arctan B_i \alpha_i, i \in \{F, R\},$$
$$F_{R,x} = C_m D - C_r - C_d v_x^2$$

where $B_i, C_i, D_i, i \in \{F, R\}, C_m$ are tire parameters, and $\alpha_i, i \in \{F, R\}$ are the front and rear slip angles, respectively. The tire parameters were identified as described in Section III-C.1, while the rolling resistance, C_r, was identified from the experiments. In addition, we incorporated the wake effect of the leading vehicle [13]. We used an adaptive drag parameter C_d, to exploit drafting effect.

2) Model Predictive Contouring Control: Motivated by [5], we adapted a MPCC scheme that accomplishes the path-following task of a vehicle in an optimal fashion via MPC. The typical MPC problem can be stated as follows:

$$\min_{\mathbf{U}} \quad J \tag{3a}$$

$$\text{s.t} \quad \mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k), \quad k = 0, ..., N-1 \tag{3b}$$

$$\mathbf{x}_k \in \mathscr{X}_{\mathscr{C}}, \mathbf{u}_k \in \mathscr{U}_{\mathscr{C}} \tag{3c}$$

$$\mathbf{x}_N \in \mathscr{X}_{\mathscr{C}N} \tag{3d}$$

$$\mathbf{x}_0 = \mathbf{x}(t_0) \tag{3e}$$

where J is the cost function, $\mathbf{U} = \{\mathbf{u}_0, ..., \mathbf{u}_{N-1}\}$ is the control sequence (i.e. the optimization variable), $\mathscr{X}_{\mathscr{C}}$ and $\mathscr{U}_{\mathscr{C}}$ are safe set for state and control, $\mathscr{X}_{\mathscr{C}N}$ is the terminal safe set, $\mathbf{x}_N$ is the terminal state and $\mathbf{x}_0$ is the state $\mathbf{x}$ at the time of measurement t_0. The safe sets are decided based on the constraints at each time step.

By extending the typical MPC problem presented above, MPCC is an algorithm that enables the vehicle to follow the path by adding the contouring cost Q and control quality cost R into the cost function. Following the definitions and notations in [5], Q and R are formulated as follows:

$$Q(\mathbf{x}_k) = q_c \hat{e}_{c,k}^2 + q_l \hat{e}_{l,k}^2 - \gamma v_{\theta,k}. \tag{4}$$

$$R(\mathbf{u}_k, \Delta\mathbf{u}_k) = \mathbf{u}_k^{\mathrm{T}} \mathbf{R_u} \mathbf{u}_k + \Delta\mathbf{u}_k^{\mathrm{T}} \mathbf{R_{\Delta u}} \Delta\mathbf{u}_k, \tag{5}$$

where $\hat{e}_{c,k}$ is the contouring error, $\hat{e}_{l,k}$ is the lag error, $v_{\theta,k}$ is the speed at time step k with respect to the reference path, and q_c, q_l, and γ are their corresponding weights.

Unlike previous studies, we directly used the future trajectories of surrounding vehicles predicted through the game-based predictor as soft constraints of the MPCC. Through this, we could avoid collisions not only with static obstacles but also with various adjacent vehicles racing with their own strategies. However, in the racing scenario, because it is a non-cooperative game, errors in predicting the opponent's driving trajectory cannot be avoided. The prediction error is generated from the uncertainty in the opponent's state estimation, vehicle model, and control law. Here, we assumed that the *combined uncertainty*, σ, can be approximated as a normal distribution $\mathscr{N}(0, diag(\sigma))$ following the independent and identically distributed conditions. Thus, we define the constraint in a finite horizon with a confidence interval p_t as follows:

$$d(\mathbf{x}_{op,k}, \mathbf{x}_k) > p_t \sigma, \quad \forall k = 0, ...N, \tag{6}$$

where the distance function d is the Euclidean distance from ego position (x_k, y_k) corresponding to $\mathbf{x}_k$, to the predicted opponent position $(x_{op,k}, y_{op,k})$ corresponding to $\mathbf{x}_{op,k}$, and p_t is the decreasing confidence interval defined for every time step.

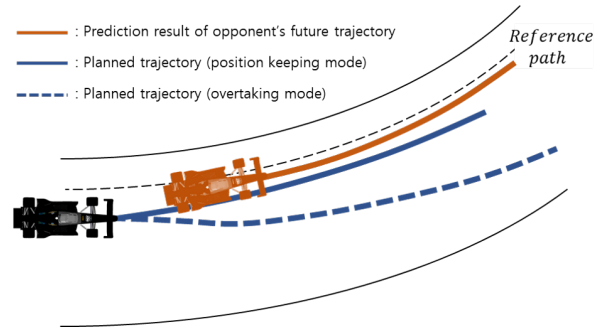The final MPCC formulation of our proposed controller

Fig. 4: Two different planner's outcomes in position-keeping mode (solid blue) and overtaking mode (dashed blue) based on the same trajectory prediction (solid orange).

with constraints can be rewritten as follows:

$$\min_{\mathbf{U}} \quad \sum_{k=0}^{N} Q(\mathbf{x}_k) + R(\mathbf{u}_k, \Delta\mathbf{u}_k) \tag{7a}$$

$$\text{s.t} \quad \mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k), \quad k = 0, ..., N-1 \tag{7b}$$

$$\mathbf{x}_0 = \mathbf{x}(t_0) \tag{7c}$$

$$\mathbf{u}_{min} \leq \mathbf{u}_k \leq \mathbf{u}_{max} \tag{7d}$$

$$\Delta\mathbf{u}_{min} \leq \Delta\mathbf{u}_k \leq \Delta\mathbf{u}_{max} \tag{7e}$$

$$(x_k - x_{ref,k})^2 + (y_k - y_{ref,k})^2 \leq R_{track}^2 \tag{7f}$$

$$d(\mathbf{x}_{op,k}, \mathbf{x}_k) > p_t \sigma. \tag{7g}$$

3) High-Level Race Strategy Planner: Professional human drivers have the ability to respond to various race circumstances while balancing their driving aggressiveness and conservativeness. To implement these racing behaviors in our controller, we hierarchically added a high-level race strategy planner before the MPCC-based low-level controller. The race strategy planner is responsible for planning the race-oriented strategies of the ego-vehicle using various configurations of the MPC controller. We used the MPC solution also in the trajectory planner implementation, with the objective function equal to (7a). We designed two distinct modes within the planner: position-keeping and overtaking. In the position-keeping mode, the contouring cost weight q_c was set to large values so that the ego-vehicle tightly follows the reference path while maximizing the aero-drafting advantage at the rear of the preceding vehicle. By contrast, the overtaking mode was configured such that the vehicle's progress is maximized by setting the speed cost weight, γ, to a large value. The two modes are selectively utilized, and the criterion for selecting the overtaking mode is whether the progress in the terminal state exceeds a certain threshold (here, we set the threshold as 3 m) compared to the predicted progress of the competitor. Fig 4 shows the difference in strategy between the two modes.

IV. EXPERIMENTS

A. Experimental Design

We used the Ansys VRXPERIENCE driving simulator, the official software of the Indy Autonomous Challenge, as shown in Fig. 5. We created a head-to-head racing scenario
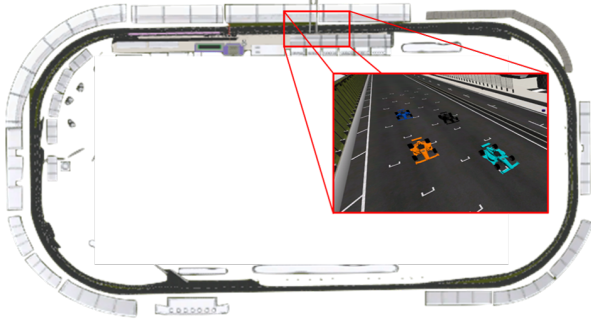
Fig. 5: Top view of the Indianapolis Motor Speedway (IMS) modeled in the Ansys VRXPERIENCE driving simulator.

with four identical vehicles in a rolling start setting and the ego-vehicle placed last in the group. Here, we set the planning horizon to 1 s, and the top speed of the vehicle was limited to 300 km/h. The position and speed of the surrounding vehicles were detected by a radar provided by the simulator. To make the experiments more realistic, we added random noise to the perception result, intentionally introducing error in the predicted trajectories.

To verify our proposed approach, we conducted the following case studies:

- Case 1) With and without the proposed high-level race strategy planner: This study was conducted in the curved segment of the track.
- Case 2) Replace the game-based predictor with the extended Kalman filter (EKF) method: This study was conducted in the straight segment of the track. For the EKF method, we assumed that the acceleration and heading did not change during the prediction.

B. Simulation Results

Fig. 6 shows the experimental results of case 1. The first row of Fig. 6 shows the trajectories of the ego and the preceding vehicles over time in a curved segment of the track. Without the race strategy planner (first row, right), the control commands were directly calculated from the MPC based low-level controller. Here, the ego-vehicle attempted to overtake the opponent to the right side at t=10. However, the throttle was released to satisfy the vehicle dynamics constraints and, accordingly, the vehicle speed decreased while the preceding vehicle ran at a constant acceleration. Because of the increased travel distance and reduced speed, the distance gap between the ego and the opponent increased. After that, the ego-vehicle returned to the reference path to reduce contouring error and maximize its progress. On the other hand, with the race strategy planner (first row, left), the ego-vehicle was driven in position-keeping mode by the planner. In this case, the ego-vehicle could accelerate more using the aero drafting advantage, but the acceleration was not enough to overtake the other. Therefore, the ego-vehicle kept the reference path driving close to the opponent and releasing the throttle, where necessary. Although there was no overtaking in both cases, there was a huge difference in
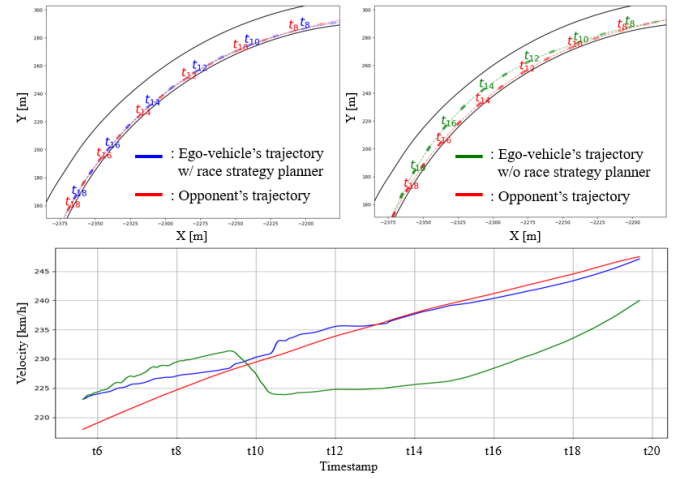

Fig. 6: Simulation results in case 1.

TABLE I: Single lap simulation results.

Predictor type	High-level race strategy planner	Collision occurred time (sec)	# of overtaking	Single lap time (sec)
EKF	W/O	t = 7.3	-	DNF
EKF	W/	t = 8.8	-	DNF
Game based	W/O	-	0	53.135
Game based	W/	-	1	52.051

terms of driving speed, as shown in the second row of Fig. 6.

Next, Fig. 7 presents the experimental results for case 2, which show the effectiveness of the proposed game-based predictor. We conducted experiments at the beginning of the race, where there are many interactions with competitors. Also, opponents' initial speed and ours were set to 100 km/h and 110 km/h, respectively. The first and the second rows of Fig. 7 show the results of the proposed predictor in action. Here, the ego-vehicle drove to the right side of the track until t=5 to reduce the contouring error with its reference path. After t=5, the ego-vehicle reacted to the predicted competitors' trajectories and drove to the left side of the track, aiming to avoid the collision without braking maneuver. On the other hand, when the EKF was applied as a predictor (third and fourth rows of Fig. 7), a collision occurred around t=5. This result clearly shows the effectiveness of our approach, that has the advantage of predicting the opponents' future trajectories based on the interaction between them.

Single lap simulation results are listed in Table I. When EKF was used as trajectory predictor, collisions occurred regardless of the presence or absence of a high-level race strategy planner, and the lap time was measured as do not finish (DNF). On the other hand, when the proposed game-based opponents' trajectory predictor was used, collision had not occurred in all cases. Furthermore, when a high-level race strategy planner was used with this predictor, 1.1 sec of lap time gain was observed.
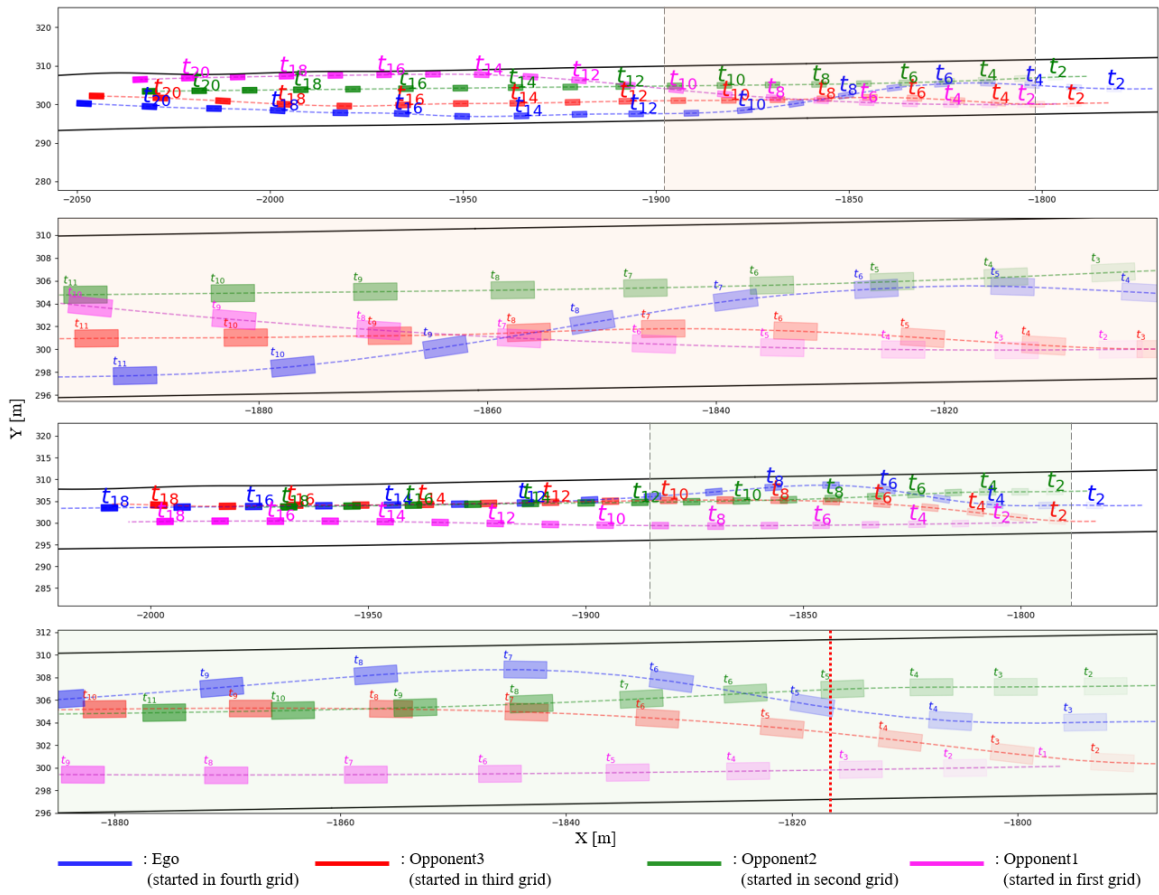
Fig. 7: Simulation results in case 2.

V. CONCLUSIONS

In this study, we introduced a game-theoretic MPC for head-to-head autonomous racing and a data-driven approach for model identification. For model identification, we explored the parameters of the highly non-linear tire model from the driving data by applying the hyperband algorithm, widely used in the machine learning field. The proposed game-theoretic MPC comprises three modules: 1) Stackelberg game-based opponents' trajectory predictor, 2) high-level race strategy planner, and 3) MPC-based low-level controller. The performance of the proposed method was demonstrated based on various scenarios in a highly realistic racing simulator. The experimental results showed that the proposed approach drives as quickly as possible while avoiding collision in a head-to-head racing situation.

We plan to expand the results of this study by adding the strategic blocking mode in the high-level race strategy planner and reinforcement learning based head-to-head autonomous driving.

REFERENCES

[1] Roborace. https://roborace.com/. Accessed: 2021-04-20.
[2] Indy autonomous challenge. https://www.indyautonomouschallenge.com/. Accessed: 2021-04-20.
[3] Darpa's robotic autonomy in complex environments with resiliency. https://www.darpa.mil/news-events/2020-10-07. Accessed: 2021-04-20.
[4] Juraj Kabzan, Miguel I Valls, Victor JF Reijgwart, Hubertus FC Hendrikx, Claas Ehmke, Manish Prajapat, Andreas Bühler, Nikhil Gosala, Mehak Gupta, Ramya Sivanesan, et al. Amz driverless: The full autonomous racing system. *Journal of Field Robotics*, 37(7):1267–1294, 2020.
[5] Alexander Liniger, Alexander Domahidi, and Manfred Morari. Optimization-based autonomous racing of 1: 43 scale rc cars. *Optimal Control Applications and Methods*, 36(5):628–647, 2015.
[6] Alexander Buyval, Aidar Gabdulin, Ruslan Mustafin, and Ilya Shimchik. Deriving overtaking strategy from nonlinear model predictive control for a race car. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2623–2628. IEEE, 2017.
[7] Mingyu Wang, Zijian Wang, John Talbot, J Christian Gerdes, and Mac Schwager. Game theoretic planning for self-driving cars in competitive scenarios. In *Robotics: Science and Systems*, 2019.
[8] Alexander Liniger and John Lygeros. A noncooperative game approach to autonomous racing. *IEEE Transactions on Control Systems Technology*, 28(3):884–897, 2019.
[9] Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. *The Journal of Machine Learning Research*, 18(1):6765–6816, 2017.
[10] Hans Pacejka. *Tire and vehicle dynamics*. Elsevier, 2005.
[11] Tong Yu and Hong Zhu. Hyper-parameter optimization: A review of algorithms and applications. *arXiv preprint arXiv:2003.05689*, 2020.
[12] Jason Kong, Mark Pfeiffer, Georg Schildbach, and Francesco Borrelli. Kinematic and dynamic vehicle models for autonomous driving control design. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, pages 1094–1099. IEEE, 2015.
[13] Alex Guerrero and Robert Castilla. Aerodynamic study of the wake effects on a formula 1 car. *Energies*, 13(19):5183, 2020.