2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

RaceMOP: Mapless Online Path Planning for Multi-Agent Autonomous Racing using Residual Policy Learning

Raphael Trumpp[1,2], Ehsan Javanmardi[2], Jin Nakazato[2], Manabu Tsukada[2], and Marco Caccamo[1]

***Abstract*— The interactive decision-making in multi-agent autonomous racing offers insights valuable beyond the domain of self-driving cars. Mapless online path planning is particularly of practical appeal but poses a challenge for safely overtaking opponents due to the limited planning horizon. To address this, we introduce RaceMOP, a novel method for mapless online path planning designed for multi-agent racing of F1TENTH cars. Unlike classical planners that rely on predefined racing lines, RaceMOP operates without a map, utilizing only local observations to execute high-speed overtaking maneuvers. Our approach combines an artificial potential field method as a base policy with residual policy learning to enable long-horizon planning. We advance the field by introducing a novel approach for policy fusion with the residual policy directly in probability space. Extensive experiments on twelve simulated racetracks validate that RaceMOP is capable of long-horizon decision-making with robust collision avoidance during overtaking maneuvers. RaceMOP demonstrates superior handling over existing mapless planners and generalizes to unknown racetracks, affirming its potential for broader applications in robotics. Our code is available at `http://github.com/raphajaner/racemop`.**

I. Introduction

Autonomous racing plays an essential role not only in creating self-driving cars [1], [2] but serves as an ideal testbed for novel planning strategies for safe autonomy [3]. Especially racing with multiple agents is interesting for decision-making methods due to its nature as a non-cooperative multi-agent system (MAS). So far, the challenging dynamic environment of racing has resulted in adopting path planners that predominantly rely on racing lines optimized offline using existing map data. For example, the Indy Autonomous Challenge 2019 winning team employed an optimal racing line combined with a graph-based module for overtaking [4].

We encourage extending the focus to mapless online path planning using only local observations from the vehicle's onboard sensors but no map data. First, a mapless planner is more flexible and easily deployed to new environments, as the environment must not be mapped beforehand. Second, as only local observations are used, it is robust to local changes in the environment as no localization is required. Results for F1TENTH competitions, a racing series for 1/10th scale autonomous RC cars [5], support this claim. In these real-world races, mapless planners often secured victories due to their robustness [6]. The same arguments apply to self-driving cars that must navigate unknown environments without map data, e.g., a rural road in a forest. Additionally, mapless planning extends the scope of autonomous racing, serving as a testbed for general robotic path planning, e.g., mobile robots typically act in non-structured mapless environments.

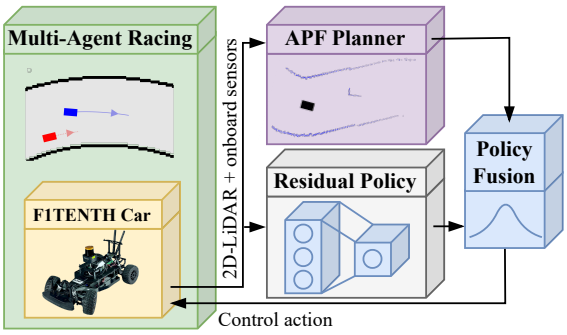


Fig. 1: Our method, named RaceMOP, is a novel mapless online path planner for multi-agent racing that uses only local observations. This method fuses an APF planner with a learned residual policy for simulated F1TENTH cars.

After achieving superhuman performance in games [7], recent works, e.g., real-world drone racing [8], have shown that deep reinforcement learning (DRL) can outperform humans also in challenging real-world decision-making problems.

This progress motivates us to propose RaceMOP, a Mapless Online Path planning method for autonomous multi-agent racing with F1TENTH cars [5]. As shown in Fig. 1, RaceMOP uses *only* 2D-LiDAR and onboard sensors but *no* map data for planning overtaking maneuvers. Our method combines an articial potential field (APF) planner, designed explicitly for multi-agent racing, with residual policy learning (RPL) to improve the vehicle's control action. It has been demonstrated that APFs are effective methods to avoid collisions for self-driving cars [9] while RPL is used successfully to improve driving policies in autonomous racing [10]–[12]. RPL trains a residual policy, parameterized by a deep neural network (DNN), with DRL to amend the output of the base policy. This approach helps to solve decision-making tasks for which DRL is data-inefficient or intractable while good but imperfect classical methods are available [13]. RaceMOP's residual policy's DNN takes stacked past observations as input, allowing long-horizon decision-making when interacting with opponents. This reasoning capability yields an advantage over classical path planners, which rely on precise localization and accurate

[1] TUM School of Engineering and Design, Technical University of Munich, Germany.
[2] Graduate School of Information Science and Technology, The University of Tokyo, Japan.
Raphael Trumpp was a JSPS International Research Fellow. Marco Caccamo was supported by an Alexander von Humboldt Professorship endowed by the German Federal Ministry of Education and Research. This research was supported by JST ASPIRE Grant Number JPMJAP2325, Japan.

predictions of future opponent trajectories and may otherwise fail to plan robust overtaking maneuvers.

Our main contributions can be summarized as follows:

- We introduce RaceMOP, a mapless online path planning method using RPL, whic outperforms classical planners in simulated multi-agent autonomous racing.
- We present a novel method for policy fusion directly in probability space, enabling unbiased bounding of action spaces essential for effective RPL.
- RaceMOP's generalization capabilities are validated through evaluation on twelve racetracks, including four previously unseen tracks. We discuss the learned driving behavior in a detailed scenario analysis.
- Our code and supplementary videos are available at `http://github.com/raphajaner/racemop`.

II. Related Work

Multi-vehicle scenarios and overtaking, e.g., on highways, are well-studied in the literature for regular self-driving cars [9]. However, it poses a different challenge to autonomous racing due to the structured environment, e.g., traffic lanes and rules [3]. For multi-agent autonomous racing, the main challenge is overtaking the non-cooperative opponents safely.

A. Map-based Methods

Access to map data allows tracking an offline-optimized racing line with an additional local planner for overtaking. Searching an offline-generated multi-layer graph model to find feasible trajectories is effective in real-world racing [4]. Raji et al. [14] use a global path as the main reference alongside a local Frenét planner for overtaking in ovals. A model predictive control method for F1TENTH cars capable of overtaking is presented by Li et al. [15].

B. Mapless Methods

Closest to our work is Zhang et al. [10], who discuss an APF planner with residual control for *single*-agent racing. Their framework does not require object detection of walls or other vehicles since they simply define each LiDAR point as an obstacle in the APF. Our approach extends this basic concept in various aspects, allowing us to focus on challenging multi-agent scenarios. We discuss further mapless methods for *multi*-agent racing in the following.

1) Classical Mapless Methods: A popular choice in F1TENTH racing is the follow-the-gap (FTG) controller introduced by Sezer et al. [16]. This reactive method steers the car toward a gap within a LiDAR reading. Similarly, the disparity extender [6] is based on detecting gaps but modifies the LiDAR's distance values to account for the vehicle's size at disparities. Another popular choice is the rapidly-exploring random tree (RRT) method and its derivatives [17]. These methods generate a tree of trajectory candidates with forward dynamics, checked for feasibility, and ranked according to their cost [18], [19]. The tentacles method [20] is an empirical approach that draws tentacles as a geometric shape in the ego vehicle's centered reference frame. Given an egocentric occupant grid, [21] proposes clothoid tentacles that are checked for feasibility. The local motion planner for the 2005 DARPA challenge used by Thrun et al. [22] is based on a similar collision avoidance method with tentacles parallel to the base path. Uusitalo and Johansson [23] present an APFs method in TORCS with multiple opponents.

2) Learning-based Mapless Methods: Loiacono et al. [24] present a controller for overtaking maneuvers in TORCS learned by reinforcement learning. While Fuchs et al. [25] show that DRL is capable of achieving superhuman performance in Gran Turismo by developing a mapless DRL controller, their model is limited to time-trial races without opponents. Song et al. [26] show that curriculum DRL is needed to learn overtaking maneuvers from LiDAR. However, their agent observes the angle and curvature of the track's centerline, and their method comes at the cost of a hand-crafted training scheme with additional hyperparameters. Moving the focus to the F1TENTH cars, Zhang and Loidl [27] present an over-taking algorithm using behavior cloning and recurrent neural network (RNN) but with access to the opponent's true state. Evans et al. [12] introduce an RPL approach for collision avoidance of static objects. A multi-team racing scenario is discussed by Werner et al. [28]. Their simulator merges an analytic model with a data-driven dynamic model to be used in the real world with a hierarchical policy structure. Schwarting et al. [29] learn a world model in latent space to imagine self-play that reduces sample generation in a multi-agent DRL training scheme. The work of Kalaria et al. [30] proposes curriculum learning to utilize a control barrier function that is gradually removed during training to not comprise the final performance.

III. Technical Background

The following section introduces the APF method for path planning and presents the DRL and RPL theory, respectively.

A. Artificial Potential Fields Planner

APF path planners are used for collision-free robot motion planning. The path is planned by defining a potential field that consists of repulsive forces for obstacles to be avoided, while the goal is defined as an attractive well [31]. Typically, a quadratic potential for repulsive forces is chosen as

$$U_{\text{rep}}(x) = \begin{cases} \frac{1}{2} k_{\text{rep}} \left(\frac{1}{\rho} - \frac{1}{\rho_0} \right)^2 & \text{if } \rho \leq \rho_0 \\ 0 & \text{if } \rho > \rho_0 \end{cases}, \tag{1}$$

where ρ is the obstacle's minimal distance to the robot's 2D position x. A threshold ρ_0 limits the area of influence, and k_{rep} is a gain factor [32]. The attractive well has the form

$$U_{\text{att}}(x) = \frac{1}{2} k_{\text{att}} \left| x - x_d \right|^2, \tag{2}$$

with k_{att} as attractive gain leading to the definition of the full potential field as $U = U_{\text{att}}(x) + \sum U_{\text{rep}}(x)$. The path is then iteratively obtained by following the potential's gradient

$$x' = x - \epsilon \frac{\nabla U}{||\nabla U||_2}, \tag{3}$$

with step size ϵ to the next position x' minimizing U.

B. Deep Reinforcement Learning

The Markov decision process (MDP) for model-free DRL is defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma)$. The stochastic action policy $\pi_\theta(a_t|s_t)$ is parameterized by θ; no forward dynamics need to be explicitly defined. This policy maps observed states $s_t \in \mathcal{S}$ to a probabilistic action space $\mathcal{P}(\mathcal{A})$ of (continuous) actions $a_t \in \mathcal{A}$. The probability for transitioning from s_t to s_{t+1} is defined by $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \mathcal{P}(\mathcal{S})$, leading to a scalar reward value r_{t+1} of the reward function $\mathcal{R}$. The discount factor is defined as γ. The goal in DRL is to find the parameters θ of the optimal policy $\pi^*_\theta(a_t|s_t)$ that maximizes the expected return $V_{\pi_\theta}(s_t) = \mathbb{E}_{\pi_\theta}\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t\right]$.

Proximal policy optimization (PPO) [33] is an *on*-policy DRL method used to learn a stochastic policy $\pi_\theta(a_t|s_t)$, where θ are the weights of a DNN. The weights θ are updated with respect to the advantage function, which requires estimating the value function $V_\phi(s_t)$, parameterized by a separate set of weights ϕ.

C. Residual Policy Learning

Silver et al. [13] introduce RPL as the combination of classical controllers with a residual policy learned by DRL.

For generality, we describe this approach by defining the action policy $\pi_\theta(a_t|s_t)$ as the result of an operation that fuses a deterministic *base* policy $\mu_{\mathrm{B}}(s_t) : \mathcal{S} \to \mathcal{A}$ with a learned stochastic residual policy $\pi_{\mathrm{R},\theta} : \mathcal{S} \to \mathcal{P}(\mathcal{A})$, parameterized by the weights θ of a DNN. Let the operator $\circledast : \mathcal{A} \times \mathcal{P}(\mathcal{A}) \to \mathcal{H}$ be defined such that the policy fusion is a mapping

$$h = \mu_{\mathrm{B}}(s_t) \circledast \pi_{\mathrm{R},\theta}(\cdot|s_t), \tag{4}$$

to parameters $h \in \mathcal{H}$ of a parameter space $\mathcal{H}$. We then define the action policy as a probability distribution $D(h, \Lambda)$ by

$$\pi_\theta(\cdot|s_t) := \mathcal{D}(h, \Lambda), \tag{5}$$

with h as the central parameter of the distribution plus possible constraints Λ. Actions a_t are sampled $a_t \sim \mathcal{D}(h, \Lambda)$ and the weights θ of policy $\pi_\theta(a_t|s_t)$ are learned by DRL.

This definition accommodates trivial expressions for a_t, including the sum $a_t = a_{\mathrm{R},t}|_{a_{\mathrm{R},t} \sim \pi_{\mathrm{R},\theta}(\cdot|s_t)} + \mu_{\mathrm{B}}(s_t)$, as used in [11], by defining D as a Dirac delta distribution, but also captures advanced fusion operators such as the truncated Gaussian used in this work. For effective learning, $D(h, \Lambda)$ should ensure that the action policy follows the base policy without bias when the residual part is off, i.e., $a_t \approx a_{\mathrm{B},t}$.

IV. Methodology

This work introduces RaceMOP, a mapless online path planner for multi-agent autonomous racing in the F1TENTH gym simulator [5], that consists of two modules (see Fig. 1):

- *Base policy* $\mu_{\mathrm{B}}(s_t)$: An APF planner with an action $a_{\mathrm{B},t}$ designed for multi-agent racing; see Sec. IV-A.
- *Residual policy* $\pi_{\mathrm{R},\theta}(a_t|s_t)$: A stochastic action policy with learnable weights θ of a DNN. The novel policy fusion with the base planner is presented in Sec. IV-B.

The methodology is developed under the following assumptions of a race environment as non-cooperative MAS:

1) An overtaking maneuver consists of only two agents.
2) Only the ego vehicle is overtaking, i.e., no defensive or adversarial behavior to defend its position is needed.
3) All opponent vehicles track a racing line in a non-reactive way, which aligns with the literature [3], [27].

A. Artificial Potential Fields Planner

The F1TENTH cars perceive their environment by a 2D-LiDAR in a fixed field-of-view (FOV) with i points $p_i = (\alpha, d)_i$ at angle α and distance d. Fig. 2 visualizes the LiDAR signal projected to the x-y-plane. All LiDAR points can be defined directly as repulsive forces in an APF for collision avoidance [10], alleviating the need to detect the opponent's position. This naive APF method fails for multi-agent racing due to the interaction with opponents and non-holonomic vehicles. We overcome these limitations as follows.

1) Down-sample Filtering: We deploy a filtering that starts from the first LiDAR point at the minimum left angle. The next point p_{i+t} gets only added to the list $P = \{p_0, ..., p_i\}$ when $||p_{i+t} - p_i||_2 > \epsilon_f$ with threshold ϵ_f. This filtering produces a smoothed x-y-representation and reduces computational resources in the following steps.

2) Closing the Gap: It rarely occurs that the APF's gradient leads to a path outside the walls. This is due to the LiDAR's limited FOV and occlusion in curves. We prevent such paths by first removing all points in distance d_f behind the car and then adding linearly interpolated artificial points between the first and last LiDAR points behind the car.

3) Goal Point Tracking: Inspired by the FTG approach, we first identify all possible gaps. A gap is described by a disparity in the LiDAR signal exceeding a threshold ϵ_d. Once all gaps are found, the candidate goal points are defined by a ray $p = (\max(d_i, d_{i+1}), (\alpha_i + \alpha_{i+1})/2)$. We select the point the furthest away from the ego as the goal point p_g.

4) Local Collision Avoidance: Instead of modeling the ego size as a point mass or circle, we define six points of the vehicle's body to calculate the APF. For each of these points, we then consider only the repulsive forces from the closest point in the filtered set of LiDAR points as visualized in Fig. 2. Experiments have shown that this improves overtaking as the car's size is correctly reflected at its sides, which is not the case when approximating the car's size as a circle.

5) Linear Potentials: Instead of the quadratic potentials as introduced in (2) and (1), we use a linear potential

$$U_{\mathrm{rep}}(x) = \begin{cases} k_{\mathrm{rep}} \left(\frac{1}{\rho} - \frac{1}{\rho_0} \right) & \text{if } \rho \leq \rho_0 \\ 0 & \text{if } \rho > \rho_0 \end{cases}, \tag{6}$$

while the attractive well has the form

$$U_{\mathrm{att}}(x) = k_{\mathrm{att}} \left| x - x_d \right|. \tag{7}$$

Our experiments show this choice helps because it keeps the attractive force constant. Otherwise, a goal point far away would lead to a stronger attractive force than a close one, which hinders local collision avoidance as the planned path typically tries to avoid the local obstacle with minimal detour in the direction of the goal. This leads to unsmooth paths that the non-holonomic F1TENTH cars cannot properly follow.

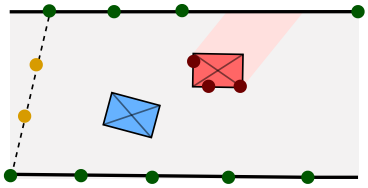


Fig. 2: The ego vehicle (blue) perceives the environment by LiDAR points; green points show the wall (black line). When another vehicle (red) is present, its shape is approximately reflected by LiDAR points (red), but parts of the wall get occluded (red area). After filtering the points, the gap behind the ego vehicle is closed with artificial points (orange).

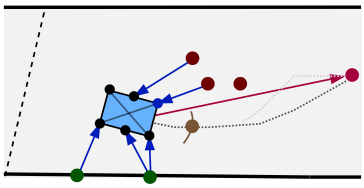


Fig. 3: Only a subset of LiDAR points (blue) closest to the ego vehicle's edge points (black) are considered as repulsive forces (blue arrows) for the APF, while the goal point (purple) is attractive. The calculated path (light gray) is smoothed (dark gray) to avoid abrupt direction changes. The tracking point (brown) is found in a fixed lookahead distance.

6) Path Planning and Smoothing: The planned path $(x, y)_{i+i}, ..., (x, y)_{i+n_p}$ is obtained by iteratively using (3) to follow the APF's gradient n_p times while the environment is assumed to be static. These raw paths are often not smooth. Additionally, APF can get stuck in local minima in symmetric passages [31], e.g., when there is only a narrow gap when overtaking, leading to a zick-zack path. We apply the filtering method described in Sec. IV-A.1 again and then further smooth the path by fitting a cubic spline. The tracking point p_t is subsequently determined from the cubic spline using the lookahead distance l_t as shown in Fig. 3.

7) Control Command with Velocity Profile: Given the tracking coordinate $p_t = (x, y)$, a pure pursuit [34] controller is used to obtain the steering command δ_t. Calculating the target velocity is challenging as it must combine the vehicle's physical properties with planning for safe overtaking. We propose to calculate the target velocity v_t as a minimum

$$v_t = \min\left(v_1(\mu_f, \delta_t), v_2(d_g)\right) \tag{8}$$

with $v_1(\mu_f, \delta_t)$ reflecting the physical friction limit [35] by

$$v_1(\mu_f, \delta_t) = \sqrt{\frac{\mu_f l g}{\tan|\delta_t|}}, \tag{9}$$

given the friction coefficient μ_f, the vehicle's wheelbase l, and the gravity constant g. The distance d_g to the goal point p_g defines $v_2(d_g)$ to model the interaction when overtaking. During overtaking, a goal point close to the ego vehicle indicates that there is no or only a narrow passage for overtaking. Therefore, $v_2(d_g)$ is proportionally decreased as it is assumed that no safe overtaking is possible.

B. RaceMOP

RaceMOP uses the presented APF planner as base policy π_B with corresponding action $a_{\mathrm{B},t}$. We implement the residual policy $\pi_{\mathrm{R},\theta}$ as a DNN that learns the parameters of a yet-to-be-defined probability distribution, namely a state-dependent parameter $\mu_{R,\theta_1} := f(s_t; \theta_1)$ and another parameter $\sigma_{R,\theta_2} := \theta_2$, i.e., the learnable parameters are $\theta = (\theta_1, \theta_2)$. The action policy's weights θ are learned by the PPO algorithm as later outlined in Sec. IV-C.

1) Policy Fusion: In reference to the arbitrary $\mathcal{D}(h, \Lambda)$ in (5), we aggregate π_R and μ_B in RaceMOP by means of a *truncated* Gaussian distribution $\mathcal{N}(\mu, \sigma, c^-, c^+)$ [36] with

$$\mu = a_{\mathrm{B},t} + \alpha \cdot \mu_{R,\theta_1} \tag{10}$$

$$\sigma = \sigma_{R,\theta_2}. \tag{11}$$

The truncation interval $[c^-, c^+]$ ensures that sampled actions are bounded as $a_t \in [c^-, c^+]$. This distribution's mean μ is also its mode. Therefore, when $\mu_{R,\theta_1} = 0$ and the mode is sampled, then $a_t = a_\mathrm{B}$. Using other distributions can lead to a biased mapping of the base action when fusing before the probability function or wrong gradients when clipping is used for bounded actions. Using the truncated Gaussian overcomes both limitations and correctly redistributes the probability mass for the truncation interval.

2) Action And State Space: RaceMOP's continues action $a_t = [v_t, \delta_t]$ consists of the target speed v_t and steering angle δ_t as the input to the car's low-level controller. Bounded actions are sampled $a_t \sim \mathcal{N}(\mu, \sigma, c^-, c^+)$ with $c^\pm = [\pm 1, \pm 1]$ as required by the low-level controller.

Only *local* information is used for the state

$$s_t^o = \begin{bmatrix} L_t & v_t^{\text{long}} & v_t^{\text{lat}} & \dot{v}_t^{\text{long}} & \dot{\psi}_t & \beta_t & \delta_t & a_{t-1} \end{bmatrix}^\top, \tag{12}$$

with the 2D-LiDAR $L_t \in \mathcal{R}^{1080}$ covering a 270° FOV. The ego's dynamic state is represented by the longitudinal velocity v_t^{long}, lateral velocity v_t^{lat}, longitudinal acceleration $\dot{v}_t^{\text{long}}$, the yaw rate $\dot{\psi}$, the slip angle β, steering δ_t, and the previous action a_{t-1}. Temporal information is embedded by stacking n_f previous states where n_s frames are skipped in between, forming the current state as $s_t = \{s^o_{t-n_f(1+n_s)}, \dots, s^o_t\}$.

3) Reward Design: Our reward is independent of map data by using the ego's longitudinal velocity to calculate the traveled distance $d_t^{\text{long}} = v_t^{\text{long}} \cdot \Delta t$ between timesteps of duration Δt as the main optimization target:

$$\begin{aligned} r_{t+1} = {} & 0.1 \cdot d_t^{\text{long}} - 0.005 \cdot |a_t - a_{t-1}| \\ & - 0.2 \cdot d_{L,t} \cdot \mathbf{1}_{d_l} + 0.5 \cdot \mathbf{1}_o - 5 \cdot \mathbf{1}_c. \end{aligned} \tag{13}$$

Large action changes between time steps are penalized, as well as small distances to obstacles with $\mathbf{1}_{d_l} = 1$ when $d_{L,t} = \min L_t < 0.4$. Successful overtaking are encouraged by $\mathbf{1}_{d_0} = 1$ and collisions are penalized $\mathbf{1}_c = 1$.

C. Learning Algorithm and Network Design

RaceMOP is based on a PPO agent [33] with action policy π_θ, value network V_ϕ, and learnable parameters $\{\phi, \theta\}$. As

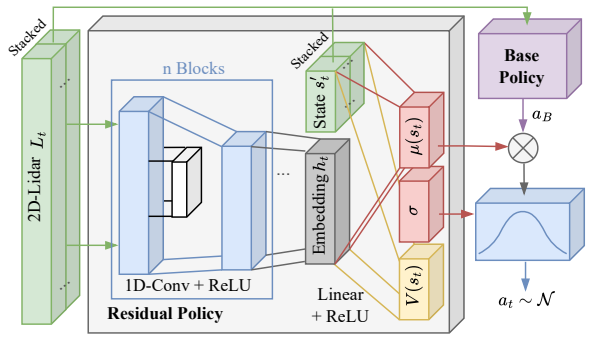


Fig. 4: RaceMOP's architecture combines a base policy with a residual policy to learn the parameters of a probability distribution $\mathcal{N}$ from only local observations $s_t = \{L_t, s'_t\}$ that contains a history of n_f frames.

PPO	Total steps	$30e^6$	Discount γ	0.99
	Learn. rate (LR)	$1e^{-4}$	GAE λ	0.95
	LR schedule	cos	Value coef.	0.5
	Traj. length	2048	Max. grad. norm	1.0
	Clip ϵ	0.1	Update epochs	7
	Batch size	512	RPO$_\alpha$ [37]	0.05
	Initialization σ_{R,θ_2}	-0.7	α (see (10)	[0.5, 0.5]
APF	Attractive coeff. k_{att}	1000	ρ_0	8.0
	Repulsive coeff. k_{rep}	25	Step size ϵ	0.1
	Lookahead target l_t	1.0	Gap distance ϵ_d	1.0
	Filtering ϵ_f	0.1	Filtering d_f	-4.0
	Iterations n_p	20		

TABLE I: Hyperparameters used in RaceMOP's modules.

shown in Fig. 4, the LiDAR signal L_t is encoded by a shared DNN consisting of $n = 5$ blocks of 1D-convolutional neural network (CNN) layers (#-filters, size, stride) with $\{(64, 6, 4), (128, 3, 2), (256, 3, 2), (256, 3, 2), (256, 3, 2))\}$, respectively, and ReLU activation. The state is built by stacking $n_f = 6$ frames to the current one, skipping $n_s = 2$ frames in between. The temporal structure in L_t is reflected by separating the different timesteps into separate input channels. The embedding h_t is obtained by a linear projection layer with 64 neurons after the 1D-CNN layers and concatenated to the remaining flattened states s'_t. Two separate network heads for the policy and value network are used, consisting of linear layers with ReLU activation and $\{256, 2\}$ and $\{256, 1\}$ units, respectively. This yields 828,293 trainable parameters overall. The policy's output is then fused with the base policy by a truncated Gaussian as discussed in Sec. IV-B.1.

V. Results

We benchmark RaceMOP in Sec. V-B and present additional results for generalization in Sec. V-C. RaceMOP's learned driving behavior is analyzed for various racing situations, including visualizations, in Sec. V-D.

A. Setup Experiments

1) Simulator Settings: We employ a custom F1TENTH gym simulator [5]. The used maps are replicas of famous real-world racetracks, appropriately downscaled for F1TENTH cars [2]. These maps feature challenging design elements like chicanes and hairpin curves. All opponents use a pure pursuit controller to follow an offline-optimized racing line with velocity profile $\{v_{w_0}, v_{w_1}, ...\} \cdot k_v$. The velocity gain k_v limits the opponent's velocity. We define 30 start positions for each racetrack. Episodes are truncated after two laps. See Sec. IV for further assumptions. The simulation is based on a single-track model with tire slip and a maximum velocity of $8.0\,\mathrm{m\,s^{-1}}$. We use a friction coefficient of $\mu_f = 0.8$ to ensure there is slip when driving too fast in curves. The observation and control frequencies are fixed to $50\,\mathrm{Hz}$, similar to real-world F1TENTH cars. For numerical stability, the simulation runs at $100\,\mathrm{Hz}$. The LiDAR data contains Gaussian noise.

2) Training Procedure: RaceMOP is trained for $30e^6$ simulation steps. Experiments have shown that high parallelization with 256 environments is beneficial. Training is randomized by racing on eight different racetracks. Additionally, $k_v \sim \{0.8, 0.75, 0.7\}$ is sampled uniformly every episode for all opponents in the same environment, and new random starting positions are chosen. Thus, the ego vehicle must learn robust overtaking strategies for various scenarios, making the task more challenging. Overtaken opponents are reset to a close position in front of the ego. Observations and rewards are normalized. We use a median filter to remove noise in the LiDAR data before being fed into the DNN. Other important hyperparameters of RaceMOP and for implementing the APF base policy can be found in Table I. Full training takes approx. 3.5 h on a server with AMD TR PRO 5975WX CPU and NVIDIA GeForce RTX 4090 GPU.

3) Evaluation Details: To analyze the RaceMOP's generalization capabilities, we evaluate our model for twelve racetracks, i.e., the eight training racetracks and four additional ones for testing. Reported results are the mean and standard deviation from five different training seeds. Each independent run is evaluated for thirty episodes with varying start positions and averaged as medians for lap times, while the other metrics are means. Nine opponents are distributed along the racetracks with a gain fixed to $k_v = 0.75$ for the evaluation. We define the following three metrics:

- $I_T[\downarrow]$: Lap time in s as duration of the running start lap.
- $I_C[\downarrow]$: The crash rate of the ego when attempting to overtake as $\frac{\#_{\text{crash}}}{\#_{\text{success}}+\#_{\text{crash}}}|_{\text{overtaking}}$ in %.
- $I_E[\downarrow]$: Environment crashes of the ego while not overtaking over the total driven distance in kilometer.

B. RaceMOP Benchmarking

This evaluation compares the performance of RaceMOP against the pure APF planner to demonstrate the effectiveness of the RPL approach. Our results are shown in Table II for the eight *training* race tracks. First, it can be seen that RaceMOP improves the base policy w.r.t. all performance metrics:

- Lab times are decreased by 8.65 % to $I_T = 52.41$ s averaged for all training racetracks.
- Despite the faster lap times, RaceMOP manages to decrease unsuccessful overtaking substantially to 0.33 %

	Racetrack Name	$I_T[\downarrow]$ in s APF	$I_T[\downarrow]$ in s RaceMOP	Δ_{rel} in %	$I_C[\downarrow]$ in % APF	$I_C[\downarrow]$ in % RaceMOP	Δ_{rel} in %	$I_E[\downarrow]$ in km^{-1} APF	$I_E[\downarrow]$ in km^{-1} RaceMOP	Δ_{abs}
Training	Budapest	57.14	$51.62^{\pm 0.45}$	−9.67	22.47	$0.4^{\pm 0.36}$	−98.22	0.0	$0.0^{\pm 0.0}$	0.0
	Catalunya	58.86	$53.67^{\pm 0.61}$	−8.81	24.05	$0.81^{\pm 0.35}$	−96.62	0.0	$0.06^{\pm 0.12}$	0.06
	Hockenheim	51.7	$47.1^{\pm 0.32}$	−8.91	24.18	$0.0^{\pm 0.0}$	100.0	0.0	$0.0^{\pm 0.0}$	0.0
	Moscow	50.0	$44.79^{\pm 1.04}$	−10.42	32.39	$0.67^{\pm 0.81}$	97.93	0.0	$0.0^{\pm 0.0}$	0.0
	Nürburgring	63.4	$57.88^{\pm 0.37}$	−8.71	24.66	$0.13^{\pm 0.29}$	99.48	0.13	$0.01^{\pm 0.02}$	−0.12
	Sakhir	62.19	$57.08^{\pm 0.31}$	−8.22	12.5	$0.53^{\pm 1.18}$	95.79	0.0	$0.0^{\pm 0.0}$	0.0
	Sepang	68.2	$62.81^{\pm 0.64}$	−7.9	23.6	$0.0^{\pm 0.0}$	100.0	0.0	$0.0^{\pm 0.0}$	0.0
	Spielberg	47.54	$44.37^{\pm 0.3}$	−6.67	29.33	$0.14^{\pm 0.31}$	99.52	0.0	$0.03^{\pm 0.03}$	0.03
	All	57.38	$52.41^{\pm 0.46}$	−8.65	23.55	$0.33^{\pm 0.19}$	−98.61	0.02	$0.01^{\pm 0.01}$	0.01

TABLE II: Performance metrics for *training* racetracks with mean and standard deviation from five random seeds for RaceMOP.

	Racetrack Name	$I_T[\downarrow]$ in s APF	$I_T[\downarrow]$ in s RaceMOP	Disparity	$I_C[\downarrow]$ in % APF	$I_C[\downarrow]$ in % RaceMOP	Disparity	$I_E[\downarrow]$ in km^{-1} APF	$I_E[\downarrow]$ in km^{-1} RaceMOP	Disparity
Test	Brands Hatch	47.89	$\mathbf{45.38^{\pm 0.36}}$	46.5	13.98	$\mathbf{0.28^{\pm 0.38}}$	21.11	0.06	$\mathbf{0.0^{\pm 0.0}}$	**0.0**
	Melbourne	65.25	$\mathbf{60.64^{\pm 0.26}}$	61.15	31.51	$\mathbf{0.28^{\pm 0.38}}$	23.38	**0.0**	$0.01^{\pm 0.02}$	0.32
	Mexico City	52.52	$47.69^{\pm 0.53}$	**47.55**	35.62	$\mathbf{0.59^{\pm 0.6}}$	29.87	**0.0**	$0.13^{\pm 0.28}$	0.29
	Sao Paulo	50.06	$\mathbf{45.87^{\pm 0.25}}$	47.72	27.14	$\mathbf{0.55^{\pm 0.58}}$	21.43	0.18	$\mathbf{0.01^{\pm 0.02}}$	0.24
	All	53.93	$\mathbf{49.89^{\pm 0.32}}$	50.73	26.21	$\mathbf{0.42^{\pm 0.34}}$	23.68	0.05	$\mathbf{0.03^{\pm 0.06}}$	0.21

TABLE III: Performance metrics for *test* racetracks with mean and standard deviation from five random seeds for RaceMOP. The best results are bold.

crashes per attempt, i.e., there is approx. one crash for 300 successful overtaking maneuvers.

- RaceMOP increases environment crashes I_E for two tracks but reduces them when the APF struggled before.

Analysis of the lap times I_C shows that the gains are similar on all training racetracks, ranging from −10.42 % for Moscow to −6.67 % for Spielberg. For the APF planner, overtaking for Moscow, a racetrack with high average curvature, is most challenging with $I_C = 32.39\,\%$. RaceMOP is capable of reducing this number substantially by −97.93 % to 0.67 %. While overtaking, most collisions of RaceMOP happen at Catalunya with $I_C = 0.81\,\%$. The best performance of RaceMOP is achieved for overtaking on the Hockenheim and Sepang racetracks where *zero* crashes are recorded.

RaceMOP's performance is overall stable for different training runs as the standard deviations of $\pm 0.46\,\%$ for I_T and $\pm 0.19\,\%$ for I_C prove. The higher deviations for I_C on Sakhir and Moscow show that if there is some instability during training for these tracks, RaceMOP compensates by improved performance on all the other racetracks. This may indicate that RaceMOP's representational capacity is limited by the used size of the DNN as performance gains on one track come at the cost of slightly more collision on another.

C. Out-of-Distribution Benchmarking

We evaluate the generalization capabilities of RaceMOP on four *test* racetracks and in comparison to another approach: a disparity extender [6] with adaptive velocity planning to avoid collisions when there is no safe gap. Our results in Table III align with the previously discussed improvements by RaceMOP. While the disparity extender can race at higher velocities[1] than the APF base planner, RaceMOP overcomes this disadvantage. These results demonstrate RaceMOP's outstanding generalization capabilities, achieving an average ratio of $I_C = 0.42\,\%$. However, the environment crashes for Mexico City increase to $I_E = 0.13\,\%$, which indicates that due to the lack of map data, there is a specific section that is not correctly interpreted by RaceMOP.

Scenario	$I_T[\downarrow]$ in s	$I_C[\downarrow]$ in %	$I_E[\downarrow]$ in km^{-1}
$v_k = 0.85$	$51.98^{\pm 0.48}$	$0.86^{\pm 0.62}$	$0.01^{\pm 0.01}$
$\texttt{clip}(a_t)$	$52.72^{\pm 0.2}$	$2.65^{\pm 0.68}$	$0.01^{\pm 0.01}$

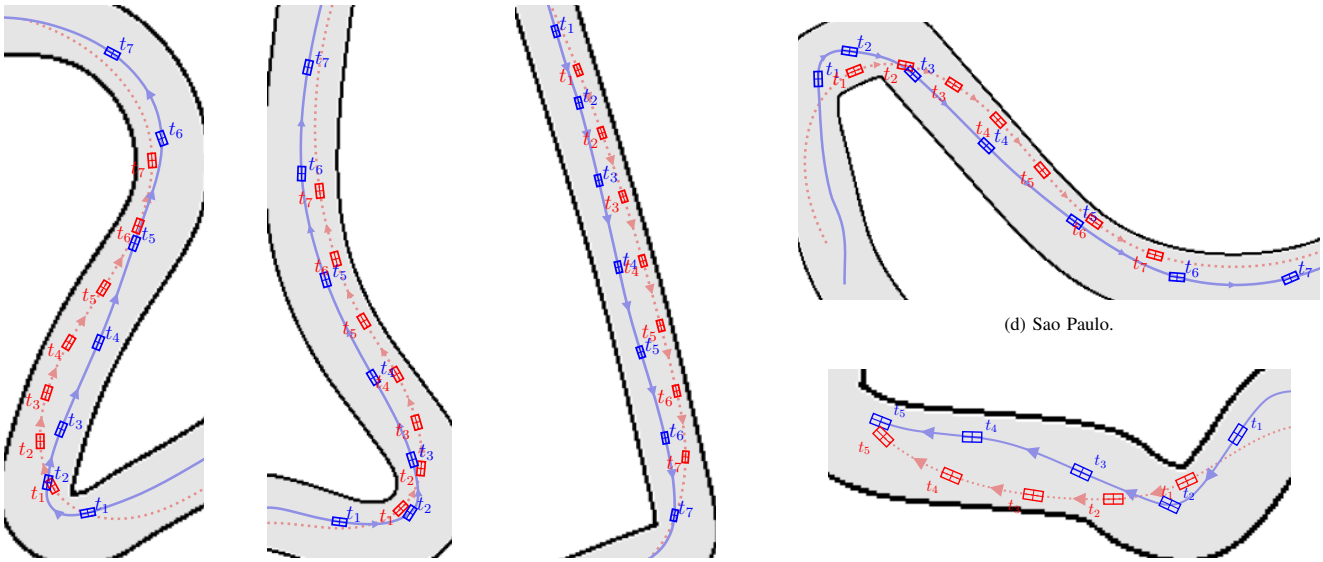
TABLE IV: Additional results for RaceMOP averaged for *training* tracks.

Table IV shows that in another out-of-distribution test with $v_k = 0.85$, RaceMOP is still capable of robust overtaking as the crash rate increases marginally to $I_C = 0.86\,\%$. These crashes occur now because RaceMOP sometimes overtakes when its velocity advantage is too small to finish the maneuvers before the opponent cuts the corner. Lap times are decreased to $I_T = 51.98\,\text{s}$ as RaceMOP loses less time following the now faster-driving opponents before overtaking.

D. Ablation Policy Fusion Method

We analyze the effect of the proposed truncated Gaussian distribution by replacing it with the clipped action sum as used in [11]. The result in Table IV shows that this fusion method can achieve lap times close to RaceMOP. However, when comparing the crash risk, it is much increased to

[1]Experiments have shown that the disparity extender as base policy does not work as well as the APF due to its rather aggressive and fast driving.



(a) Nürburgring. (b) Sepang. (c) Melbourne. (d) Sao Paulo. (e) Moscow (crash).

Fig. 5: Exemplary overtaking maneuvers of RaceMOP for five different, replicated real-world racetracks where the ego vehicle (blue, full line) overtakes the opponent (red, dashed line), showing various strategic behaviors. Discrete timesteps $t_1, ..., t_7$ of the vehicle's pose are given every 0.5 s.

$I_C = 2.65\,\%$ from $0.33\,\%$. This experiment demonstrates the substantial advantage of our proposed policy fusion method.

E. Scenario Analysis

We discuss RaceMOP's behavior with five examples. Our analysis of various scenarios and racetracks shows that RaceMOP's advantage is a robust overtaking behavior at curves. Typically, RaceMOP will approach a curve fast, break if the opponent switches sides when cutting the curve, and then accelerate quickly to pass the opponent. This behavior is effective, affirming that RaceMOP is capable of long-horizon reasoning from local observations where other approaches typically require map data.

1) Inside Overtaking (Fig. 5a and Fig. 5b): The ego vehicle waits with the attempt as the opponent cuts the corner. After the apex, the overtaking becomes feasible as a gap opens at the inside, and the ego accelerates to overtake the opponent at a safe distance.

2) High Velocity Overtaking (Fig. 5c): The ego uses its velocity advantage to overtake the opponent before a 90° curve by breaking late, requiring a careful estimation of its physical driving limit and the opponent's trajectory to finish the maneuver in time.

3) Outside Overtaking (Fig. 5d): The opponent follows the wall in a curve closely. RaceMOP has learned to wait first and then takes advantage of this behavior, passing the opponent on the outside to finish the maneuver.

4) Crash (Fig. 5e): This unsuccessful overtaking happens at a difficult section where RaceMOP first waits with the overtake. It then starts overtaking but incorrectly interprets the opponent's behavior, who cuts the corner. As the opponent closes in, the ego is not fast enough to finish the maneuver, leading to a crash.

VI. Limitations

While RaceMOP shows strong abilities for generalization and can even overtake fast-driving opponents in out-of-distribution settings, we assume that the opponents do not show active defensive behavior nor that RaceMOP is surpassed. While these assumptions align with the current literature, more various opponent behaviors should be evaluated in future work, e.g., multi-agent DRL with self-play.

For the time being, the difficult multi-agent setup prevents real-world experiments. First, a large racetrack with enough space for two cars side-by-side is required. Second, real-world training is impossible for the used $30e^6$ interactions. Thus, a direct sim-2-real deployment is required. RaceMOP's robust performance in simulation holds the promise that we can bridge the sim-2-real gap efficiently by using additional domain randomization and finetuning with real-world data. However, the current software stack does not incorporate such methods, preventing reliable real-world deployment.

VII. Conclusion

We presented RaceMOP, a map-less online path planner for multi-agent autonomous racing. RaceMOP combines an APF planner with a learned residual policy. Our results demonstrate that RaceMOP clearly outperforms comparable, map-less planners with a collision ratio of $I_C = 0.33\,\%$ and $I_C = 0.42\,\%$ for training and test racetracks, respectively. A key component of RaceMOP's strong performance is the presented novel method for policy fusion in probability space that uses a truncated Gaussian distribution. RaceMOP's advantage is its ability to make long-horizon decisions despite lacking map data, where naive planners fail.

Our future work includes testing RaceMOP against various opponent strategies and eventually transferring our method to a real-world race with multiple F1TENTH cars. Additionally, we plan an ablation study on RaceMOP's architecture as incorporating an RNN can improve training efficiency and performance for interaction-rich environments [38]. Another interesting aspect will be to incorporate continuous action-mapping techniques for safe learning [39] into our method.

REFERENCES

[1] A. Wischnewski, M. Geisslinger, J. Betz, T. Betz, F. Fent, A. Heilmeier, L. Hermansdorfer, T. Herrmann, S. Huch, P. Karle *et al.*, "Indy autonomous challenge-autonomous race cars at the handling limits," in *12th International Munich Chassis Symposium 2021*. Springer, 2022, pp. 163–182.

[2] J. Betz, H. Zheng, A. Liniger, U. Rosolia, P. Karle, M. Behl, V. Krovi, and R. Mangharam, "Autonomous vehicles on the edge: A survey on autonomous vehicle racing," *IEEE Open Journal of Intelligent Transportation Systems*, vol. 3, pp. 458–488, 2022.

[3] V. Suresh Babu and M. Behl, "Threading the needle—overtaking framework for multi-agent autonomous racing," *SAE International Journal of Connected and Automated Vehicles*, vol. 5, no. 1, 2022.

[4] T. Stahl, A. Wischnewski, J. Betz, and M. Lienkamp, "Multilayer graph-based trajectory planning for race vehicles in dynamic scenarios," in *Int. Conf. on Intelligent Transportation Systems*. IEEE, 2019.

[5] M. O'Kelly, H. Zheng, D. Karthik, and R. Mangharam, "F1tenth: An open-source evaluation environment for continuous control and reinforcement learning," in *NeurIPS 2019 Competition and Demonstration Track*. PMLR, 2020, pp. 77–89.

[6] N. Otterness, "The "Disparity Extender" Algorithm, and F1/Tenth." [Online]. Available: https://www.nathanotterness.com/2019/04/the-disparity-extender-algorithm-and.html

[7] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[8] E. Kaufmann, L. Bauersfeld, A. Loquercio, M. Müller, V. Koltun, and D. Scaramuzza, "Champion-level drone racing using deep reinforcement learning," *Nature*, vol. 620, no. 7976, pp. 982–987, 2023.

[9] P. Lin, W. Y. Choi, and C. C. Chung, "Local path planning using artificial potential field for waypoint tracking with collision avoidance," in *Int. Conf. on Intelligent Transportation Systems*, 2020, pp. 1–7.

[10] R. Zhang, J. Hou, G. Chen, Z. Li, J. Chen, and A. Knoll, "Residual policy learning facilitates efficient model-free autonomous racing," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11 625–11 632, 2022.

[11] R. Trumpp, D. Hoornaert, and M. Caccamo, "Residual policy learning for vehicle control of autonomous racing cars," in *IEEE Intelligent Vehicles Symposium*. IEEE, 2023, pp. 1–6.

[12] B. Evans, H. A. Engelbrecht, and H. W. Jordaan, "Learning the subsystem of local planning for autonomous racing," in *2021 20th International Conference on Advanced Robotics (ICAR)*. IEEE, 2021, pp. 601–606.

[13] T. Silver, K. Allen, J. Tenenbaum, and L. Kaelbling, "Residual policy learning," *arXiv preprint arXiv:1812.06298*, 2018.

[14] A. Raji, A. Liniger, A. Giove, A. Toschi, N. Musiu, D. Morra, M. Verucchi, D. Caporale, and M. Bertogna, "Motion planning and control for multi vehicle autonomous racing at high speeds," in *Int. Conf. on Intelligent Transportation Systems*. IEEE, 2022.

[15] N. Li, E. Goubault, L. Pautet, and S. Putot, "A real-time nmpc controller for autonomous vehicle racing," in *2022 6th International Conference on Automation, Control and Robots (ICACR)*. IEEE, 2022, pp. 148–155.

[16] V. Sezer and M. Gokasan, "A novel obstacle avoidance algorithm: "follow the gap method"," *Robotics and Autonomous Systems*, vol. 60, no. 9, pp. 1123–1134, 2012.

[17] S. Karaman and E. Frazzoli, "Incremental sampling-based algorithms for optimal motion planning," *Robotics: Science and Systems*, vol. 104, no. 2, pp. 267–274, 2010.

[18] S. Feraco, S. Luciani, A. Bonfitto, N. Amati, and A. Tonoli, "A local trajectory planning and control method for autonomous vehicles based on the rrt algorithm," in *AEIT international conference of electrical and electronic technologies for automotive*. IEEE, 2020, pp. 1–6.

[19] L. Ma, J. Xue, K. Kawabata, J. Zhu, C. Ma, and N. Zheng, "A fast rrt algorithm for motion planning of autonomous road vehicles," in *Int. Conf. on Intelligent Transportation Systems*. IEEE, 2014, pp. 1033–1038.

[20] F. Von Hundelshausen, M. Himmelsbach, F. Hecker, A. Mueller, and H.-J. Wuensche, "Driving with tentacles: Integral structures for sensing and motion," *Journal of Field Robotics*, vol. 25, no. 9, 2008.

[21] C. Alia, T. Gilles, T. Reine, and C. Ali, "Local trajectory planning and tracking of autonomous vehicles, using clothoid tentacles method," in *IEEE Intelligent Vehicles Symposium*. IEEE, 2015, pp. 674–679.

[22] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann *et al.*, "Stanley: The robot that won the darpa grand challenge," *Journal of Field Robotics*, vol. 23, no. 9, pp. 661–692, 2006.

[23] T. Uusitalo and S. J. Johansson, "A reactive mutli-agent approach to car driving using artificial potential fields," in *IEEE Conference on Computational Intelligence and Games*. IEEE, 2011, pp. 203–210.

[24] D. Loiacono, A. Prete, P. L. Lanzi, and L. Cardamone, "Learning to overtake in torcs using simple reinforcement learning," in *IEEE Congress on Evolutionary Computation*. IEEE, 2010, pp. 1–8.

[25] 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.

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

[27] J. Zhang and H.-W. Loidl, "F1tenth: An over-taking algorithm using machine learning," in *International Conference on Automation and Computing*. IEEE, 2023, pp. 01–06.

[28] P. Werner, T. Seyde, P. Drews, T. M. Balch, I. Gilitschenski, W. Schwarting, G. Rosman, S. Karaman, and D. Rus, "Dynamic multi-team racing: Competitive driving on 1/10-th scale vehicles via learning in simulation," in *Conf. on Robot Learning*, 2023.

[29] W. Schwarting, T. Seyde, I. Gilitschenski, L. Liebenwein, R. Sander, S. Karaman, and D. Rus, "Deep latent competition: Learning to race using visual control policies in latent space," in *Conf. on Robot Learning*. PMLR, 2021, pp. 1855–1870.

[30] D. Kalaria, Q. Lin, and J. M. Dolan, "Towards optimal head-to-head autonomous racing with curriculum reinforcement learning," *arXiv preprint arXiv:2308.13491*, 2023.

[31] M. G. Park, J. H. Jeon, and M. C. Lee, "Obstacle avoidance for mobile robots using artificial potential field approach with simulated annealing," in *IEEE International Symposium on Industrial Electronics Proceedings*, vol. 3. IEEE, 2001, pp. 1530–1535.

[32] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *Int. Conf. on Robotics and Automation*, vol. 2, 1985, pp. 500–505.

[33] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[34] R. C. Coulter *et al.*, *Implementation of the pure pursuit path tracking algorithm*. Carnegie Mellon University, The Robotics Institute, 1992.

[35] B. D. Evans, H. W. Jordaan, and H. A. Engelbrecht, "Safe reinforcement learning for high-speed autonomous racing," *Cognitive Robotics*, vol. 3, pp. 107–126, 2023.

[36] J. Burkardt, "The truncated normal distribution," *Department of Scientific Computing Website, Florida State University*, vol. 1, p. 35, 2014.

[37] M. M. Rahman and Y. Xue, "Robust policy optimization in deep reinforcement learning," *arXiv preprint arXiv:2212.07536*, 2022.

[38] R. Trumpp, M. Büchner, A. Valada, and M. Caccamo, "Efficient learning of urban driving policies using bird's-eye-view state representations," in *Int. Conf. on Intelligent Transportation Systems*, 2023, pp. 4181–4186.

[39] M. Theile, D. Bernardini, R. Trumpp, C. Piazza, M. Caccamo, and A. L. Sangiovanni-Vincentelli, "Learning to generate all feasible actions," *IEEE Access*, vol. 12, pp. 40 668–40 681, 2024.