Residual Policy Learning Facilitates Efficient Model-Free Autonomous Racing

Ruiqi Zhang[1], *Student Member, IEEE,* Jing Hou[1], Guang Chen[1,2*], *Member, IEEE,* Zhijun Li[3], *Fellow, IEEE,* Jianxiao Chen[1] and Alois Knoll[2], *Senior Member, IEEE,*

***Abstract*—Autonomous racing is a challenging task due to the safety requirement while driving aggressively. Most previous solutions utilize the prior information or depend on complex dynamics modeling. Classical model-free reinforcement learning methods are based on random sampling, which severely increases the training consumption and undermines the exploration efficiency. In this paper, we propose an efficient residual policy learning method for autonomous racing (ResRace), which leverages only the real-time raw observation of LiDAR and IMU for low-latency obstacle avoiding and navigation. We first design a controller based on the modified artificial potential field (MAPF) to generate a policy for navigation. Besides, we utilize the deep reinforcement learning (DRL) algorithm to generate a residual policy as a supplement to obtain the optimal policy. Concurrently, the MAPF policy effectively guides the exploration and increases the update efficiency. This complementary property contributes to the fast convergence and few required resources of our method. We also provide extensive experiments to illustrate our method outperforms the leading algorithms and reaches the comparable level of professional human players on the five F1Tenth tracks.**

***Index Terms*—Autonomous racing, deep reinforcement learning, artificial potential field**

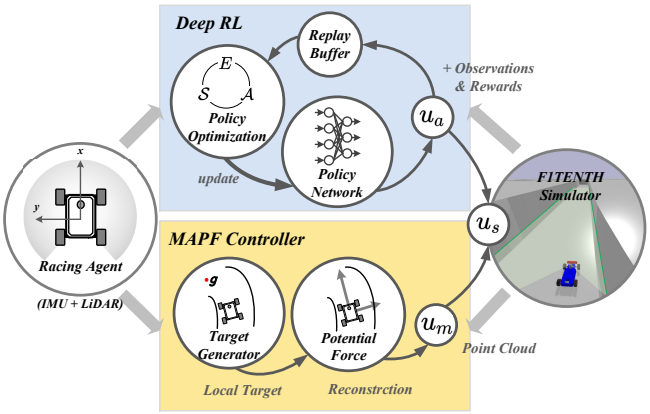

Fig. 1. The overview of ResRace. Our framework mainly consists of two parallel action generators. The MAPF controller provides a fundamental action u_m through the target generator and defined potential field to guide the exploration. The DRL agent provides a residual action u_a to optimize the summary action $u_s = u_m + u_a$ and minimize the lap time.

I. Introduction

AUTONOMOUS racing is a promising issue and has obtained much attention from research institutions and enterprises in the last decades. The objective of racing players is to complete the laps as fast as possible. The players are required to generate precise actions and aggressively drive at the dynamics limitation of vehicles. To solve this problem, classical approaches decouple autonomous racing into trajectory planning and controller optimization [1]–[3]. These approaches are widely studied and show impressive results with optimization-based techniques and model predictive control. However, their performances are highly related to the selection of parameters, and the reference trajectory requires prior information like fine dynamics model, global maps and known routes. Meanwhile, they require expensive hardware for nonlinear optimization and prediction, which undermine the economy of application for autonomous racing.

To handle the complex nonlinear dynamics models, some researchers optimize the control strategy through real-world data sets and develop the learning-based frameworks [4]–[6]. Through expert demonstrations and labeled data, neural networks can implicitly construct the mapping between raw observations and control strategy. Although these methods can effectively overcome the real-time and adaptability constraints of conventional approaches, they heavily rely on the graphics processing unit (GPU) and impose much higher hardware requirements [7]. Besides, their performances are limited by the quality and quantity of data sets so they can hardly outperform the human players.

Recently, deep reinforcement learning (DRL) [8] is considered as a promising solution for robot control and policy optimization [9]–[12]. Concurrently, many works demonstrate the outstanding performance of DRL for end-to-end autonomous driving and racing [13]–[16]. Classical DRL methods utilize the Gaussian probability distributions to cover workable action space. Nevertheless, these DRL frameworks explore and are initialized randomly so their training processes are always time-consuming. Moreover, to handle the high-dimensional observation, DRL frameworks require feature selection [16] or observation encoders [14], [17] to squeeze their observation into a latent feature, while these techniques cause information loss and undermine their generation ability.

To solve existing problems, we are inspired by our driving experiences, where drivers pay more attention to the local objects like the visible obstacles or path edges. These observations directly guide the driving route, velocity and direction. Meanwhile, we notice experiences of human drivers can be generalized to various vehicular dynamics and scenarios, while they are not dependent on the known dynamics model. Thus,

*Correspondence to Guang Chen, Email: guangchen@tongji.edu.cn
[1]Tongji University, Shanghai 200092, China
[2]Technical University of Munich, Munich 80333, Germany
[3]University of Science and Technology of China, Hefei 230026, China

in this paper, we propose a novel algorithm for autonomous racing with DRL and modified artificial potential field (MAPF) method named ResRace. It is model-free and only utilizes the real-time observations of a 2D LiDAR and an inertial measurement unit (IMU) to achieve efficiency and low-latency control. We first leverage the raw LiDAR observation to select a local target. Then the MAPF-based controller takes the point cloud and the local target to generate a fundamental action, which guides the exploration and provides better experiences for training. Concurrently, the policy network is updated and generates a supplemental action. Then the racing agent adopts the sum of the above two actions. In brief, our main contributions are three-fold:

- We propose an efficient model-free algorithm ResRace for autonomous racing, which is not dependent on prior information and utilizes only the real-time observation of sensors. Besides, our method obtains better real-time performance than other algorithms due to a more concise pipeline and fewer parameters.
- We illustrate the complementary property of crucial modules in ResRace and solve the inefficient exploration problem of model-free approaches. ResRace can be easily trained and converges in much fewer time-steps than other leading approaches.
- We provide extensive experiments to illustrate our method outperforms the existing leading baselines and reaches the comparable level of professional human players. The validation results show ResRace is robust enough to handle the different scenarios and system dynamics.

II. Related Works

The autonomous driving task has been widely studied in the past decades. Here, we divide previous studies by their methodology and they can be separated into three groups: classical hierarchical control approaches, supervised learning approaches, and reinforcement learning approaches.

Classical Control. Most previous researches describe autonomous driving as a hierarchical perception-planning-control task. In this paradigm, model predictive control (MPC) is widely used and demonstrates outstanding results in motion control [18], [19]. As an improved version, the MPC with Gaussian Process (GP)-based optimized dynamics models is a practical method for autonomous racing [20], [21]. With the fine dynamics model and the optimized reference trajectories, these methods show excellent performances in both simulation and real-world. While the deficiency on flexibility and adaption prompts researchers to focus on the learning-based MPC [22]–[24]. By generating vehicle models and control strategies through the neural network, learning-based MPC can be utilized in diverse scenarios. Nevertheless, an inevitable dilemma is the trade-off between computation consumption and performance.

Deep and Imitation Learning. To skip modeling process, researchers develop the learning-based end-to-end methods to generate control policy directly from observation. The deep neural network possesses impressive ability in feature extraction and pattern recognition are impressive so that it can be deployed on autonomous vehicle [4], [25], [26]. As a symbolic work, researchers successfully develop a convolutional neural network (CNN)-based controller to follow the road and lane [5]. Similarly, imitation learning leverages expert demonstration as a template and is proved to be feasible in off-road self-driving [6]. However, though they overcome the adaptability constraints, their performance is highly depends on the quantity and quality of labeled data and they can hardly outperform the human [27].

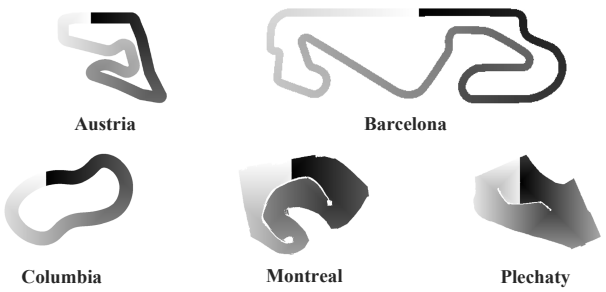

Fig. 2. The bird's-eye view of relative progress tracks in F1Tenth simulator [31]. These tracks have different characteristics in length, width and curves, which makes our racing tasks more challenging.

Reinforcement Learning. Instead of establishing the data sets, classical reinforcement learning (RL) collects experiences and updates the policy by continuous intersection with dynamic environment. Prior works prove RL is an efficient solution for various complex tasks [13]–[16], [28]. Broadly, reinforcement learning is grouped into model-based RL and model-free RL according to whether agent establishes the transition model. Many researches prove that model-free RL is practicable in realistic racing games [13], [14], [16], but the stochastic exploration could damage robots and hinders their applications in the real-world. Meanwhile, random initialization and sampling also cause their unstable training process and extensive time consumption. On the contrary, in Racing-Dreamer [15], researchers introduce a model-based method Dreamer [29] to learn the system dynamics from interactions and construct latent imagination of future state. However, this approach requires expert demonstration to pre-train and the imagination mechanism severely increase the computational and time consumption. Besides, the performance of model-based RL methods is significantly determined by the model accuracy [30]. In summary, though DRL achieves outstanding performance in both real and simulated scenarios, we still need a more robust and efficient solution for aggressive but safe autonomous racing.

III. Preliminaries

In this section, we explain the crucial definitions and the settings of the racing agent and simulation environment, which are the premise of our methodology in subsequent sections.

Problem Definition. We formalize autonomous racing as a partially observable Markov decision process (POMDP). The POMDP can be presented as a tuple $(\mathcal{S}, \mathcal{A}, \Omega, \mathcal{O}, \mathcal{T}, \mathcal{R})$, where Ω and $\mathcal{R}$ are respectively the observation and reward. We will discuss our reward design and observation

settings in the subsequent sections. $s \in \mathcal{S}$ is the possible state for racing agent and $u \in \mathcal{A}$ is the possible action. $\mathcal{O} : \mathcal{S} \times \Omega \rightarrow [0, 1]$ includes the system uncertainty and represent the probability of observation in a given state. $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$ is the transition function under a specific state-action and also includes the uncertainty of system. For the exploration process, a behavior trajectory can be presented as $\{s_0, u_0, r_0, s_1, u_1, r_1, \cdots, s_t, u_t, r_t, \cdots\}$, where t is the time-step and r_t is the instant reward provided by $\mathcal{R}$ at the time t.

Simulation Environment. In this paper, the MAPF controller and DRL agent utilize the real-time observation LiDAR and IMU. We leverage an PyBullet-based [32] F1Tenth environment for simulation [31]. We deploy the agents on the 5 tracks for evaluation with different difficulties as shown in Fig. 2. For example, Barcelona consists of long straights and bends, while Montreal and Plechaty have changing widths and irregular edges. The diversity of tracks makes the racing task more challenging and ensures the reliability of our conclusion.

Agent Settings. In the F1Tenth simulator, the agent is a rigid body with URDF models. The racing agent is equipped with a 2D LiDAR with a maximum 10 meters range and 675 measurements evenly distributed over a 270° field of view. Besides, a 60Hz IMU measures the motion of the agent, including the transverse and longitudinal velocity and acceleration in the horizontal plane. The action of agent u_a can be described as $u_a = \{\alpha, T\}$, where α denotes the steering angle in $[-45°, +45°]$ and T denotes the motor torque.

IV. Methodology

Based on the preliminaries, we illustrate the complementary property between the MAPF controller and DRL module and the pipeline of ResRace in this section. Additionally, we explain the principles and methodology of the two modules in detail to emphasize their contributions.

A. Principle of ResRace

The racing agent is required to avoid the collision with the track edge and complete the laps in a given direction in the F1Tenth simulator. Meanwhile, to guarantee the safety and process of high-frequency information, our method should respond with low latency. We consider that artificial potential field (APF) [33] is an intuitive and efficient method for autonomous navigation [34]–[36]. The inner logic of APF is digestible: the agent repulses the obstacle and is attracted by its target so that it can avoid the collision and reach the goal. Particularly, when the agent is close to obstacles, APF generates a large repulsion force and effectively guarantees safety. However, though its principle determines it is smoother and safer than trajectories provided by other approaches, the control policy of APF is sub-optimal in most cases [37]. To obtain the optimal policy, we provide a residual policy as a supplement of the APF policy so a parallel DRL module is introduced. Random exploration extends the training time and increases the risk [38] in conventional DRL methods, while previous research [39] proves that providing a prior workable policy for exploration improves the performance and accelerates the convergence effectively. Meanwhile, APF has low complexity and does not require the dynamics model so it is suitable for cooperating with model-free DRL algorithms. Thus, APF policy can be utilized as the prior policy to guide the exploration, and the DRL module provides a supplementary policy to optimize the APF policy. Their complementarity ensures the validity and outstanding performance of ResRace.

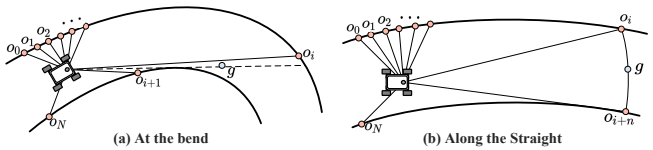

Fig. 3. The demonstration of local target selection when the racing agent drive (a) at the bend and (b) along the straight. The point cloud can be regarded as a set of tiny obstacles $\{o_0, o_1, o_2, \cdots, o_N\}$.

Pipeline of ResRace. As shown in Fig. 1, ResRace consists of two main modules: an MAPF controller and a DRL module. The MAPF controller takes only the real-time point clouds as its input and outputs a fundamental action u_m. Each point in the point cloud is regarded as a tiny obstacle, and the target generator takes the point cloud and generates a local target. According to the position of local target and obstacles, the action $u_m \in [-1, +1]$ is calculated by the defined potential field and policy function. Concurrently, the policy network, a multi-layer perceptron (MLP) is randomly initialized and provides an action u_a. The u_a is set in $[-2, +2]$ to ensure it can completely veto the u_m. The racing agent takes the truncated sum of actions u_s = *clip* $(u_m + u_a,\ -1,\ +1)$ to drive. During training, the replay buffer stores the observation from LiDAR and IMU, reward and the residual action u_a. The policy optimization algorithm samples from the replay buffer and updates the network.

B. MAPF Controller

Target Generator. The MAPF controller first utilizes raw observation of LiDAR to generate a local target g. As an intuitive demonstration in Fig. 3(a), when the agent drives at bends, the detected edge switches between the point o_i and o_{i+1}, causing the difference of adjacent distance measurements D_o^i and D_o^{i+1} are greater than the track width. In this case, the center point of segment $o_i o_{i+1}$ is selected as the local target of the MAPF controller. The other case is agent driving along the long straight as shown in Fig. 3(b). On the long straight, there is an observation arc $\{o_i, \cdots, o_{i+n}\}$ reaching the limitation range of LiDAR, which means the next bend is out of detection range. Thus, we select the center point $o_{\lfloor i+\frac{n}{2} \rfloor}$ of the arc as the attractive target.

$$U_{att} = \frac{1}{2} D_g^2, \quad U_{rep}^j = \frac{1}{2} D_o^j \left(1/D_o^j + 1/D_e\right)^2 \tag{1}$$

Potential Field Definition. The attractive and repulsive potential field function is defined as Equation (1). U_{att} is the attractive potential of the local target g. D_g is the relative position of the local target. Similarly, U_{rep}^j presents the repulsive potential of the j-th point in the point cloud and D_o^j presents its distance. D_e is the effective range of repulsive potential

and is set as 2 meters. Based on the above field functions, the potential force can be calculated as Equation (2)-(4). The constant $\eta = 0.1$ is a ratio coefficient to balance the attractive and repulsive potential forces.

$$F_{att}^{g} = - \bigtriangledown U_{att}^{g} \tag{2}$$

$$F_{rep}^{j} = - \bigtriangledown U_{rep}^{j},\ s.t.\ D_e \leq D_o^j \tag{3}$$

$$u_m = tanh\ \left(F_{att}^{g} \cdot + \eta \sum\nolimits_{j=1}^{n} F_{rep}^{j} \cdot \mathcal{X}_{xy} \right) \tag{4}$$

Modification in APF. The stagnation at dead points is a common issue for APF [37]. To solve this problem, we set a probability $\gamma = 0.1$ for the agent to explore forward with a tiny velocity when the stagnation occurs. Furthermore, due to the effective range of LiDAR is 270°, the detected points are disproportional on the x-axis and y-axis so that we set a vector $\mathcal{X}_{xy} = [0.05, 0.02]^T$ to scale the force and eliminate the imbalance of potential force. Then the hyperbolic tangent function transfers potential forces to the action in the range $[-1, +1]$. The empirical results show that this modification significantly contributes to ResRace performance and we will discuss in the experimental section.

C. Reinforcement Learning Module

Residual Policy Optimization. The policy optimization method is replaceable in the DRL module and in this paper, we utilize two classic online model-free policy optimization algorithms TRPO [40] and PPO [41]. As an actor-critic style description, the residual action u_a is provided by the actor network $\pi(s \mid \theta)$ with trainable parameters θ. Meanwhile, the critic network $V(s \mid \theta_v)$ is parameterized by θ_v and estimates the value of state. The critic network is trained with the mean square error $L_v(v_{ture}, v_e)$ between the estimation value v_e and the true one v_{ture}. The probability ratio is defined as $p_\theta = \pi(s, u_a)/\pi_{old}(s, u_a)$ to describe the similarity of the old policy π_{old} and the updated one π. The generative advantage function [42] $\hat{A}$ is introduced to reduce the error of the advantage estimation.

$$L_{trpo} = \hat{\mathbb{E}}(p_\theta \hat{A}),\ s.t.\ KL\left(\pi_{old}, \pi\right) \leq \delta \tag{5}$$

$$L_c(\theta) = \hat{\mathbb{E}}\ \left[min(p_\theta \hat{A}),\ clip\left(p_\theta, 1 - \epsilon, 1 + \epsilon\right) \right] \tag{6}$$

$$L_{ppo} = \hat{\mathbb{E}}\ \left[L_c(\theta) - L_v + \mathcal{H}(\pi) \right] \tag{7}$$

In TRPO [40], the KL divergence is utilized to constrain the policy update with the divergence threshold $\delta = 0.01$, and its surrogate objective is maximizing Equation (5). While in PPO [41], the policy update is constrained by the clipped objective as Equation (6) with the ratio $\epsilon = 0.2$. The L_v is the mean square error of value estimation. Besides, we set the policy entropy $\mathcal{H}(\pi)$ as suggested in prior works [43], [44] to ensure sufficient exploration and the surrogate objective of PPO is maximizing the Equation (7).

$$\mathcal{R} = \mathcal{R}_{fin} + \mathcal{R}_a\left(u\right) - \mathcal{P}_a\left(u\right) - \mathcal{P}_s \tag{8}$$

Reward Design. The racing task is minimizing the lap time, which is equivalent to maximizing the lap progress in the given time-steps. For DRL method, its objective is to generate a policy to maximize the episode return. Hence, the reward function should present the intention of minimizing the lap time. Here, we define the reward function in ResRace as Equation (8). The discrete reward $\mathcal{R}_{fin} = +100$ is issued when passing the finish line to encourage the agent to complete more laps in one episode. However, this sparse reward signal is difficult to attribute to specific actions, we thus add three continuous rewards. The continuous reward $\mathcal{R}_a\left(u\right)$ linearly depends on the agent velocity to encourage aggressive driving. The penalty $\mathcal{P}_a\left(u\right)$ linearly depends on the action difference of two adjacent time-steps and the absolute value of actions. It significantly restrains undesired bang-bang control of DRL methods [45], [46]. In racing, unnecessary and excessive actions cause a longer racing path and speed reduction. Meanwhile, the safety penalty $\mathcal{P}_s = 1$ restrains the collision with the track edge. Our reward definition can obviously accelerate the convergence and improve final performance, which will be demonstrated in the ablation experiment section. More methodology details can be found at *github.com/ispc-lab/resrace*.

V. Experimental Setup

In this section, we introduce the existing leading baselines and experimental settings. Additionally, we transfer our method to unseen tracks to demonstrate its generation ability. Besides, we illustrate its dynamics robustness and low parameter sensitivity. All experiments are conducted with INTEL Gold 5218R CPU and NVIDIA GeForce RTX 2070S GPU.

Baselines. RacingDreamer [15] discusses the state-of-the-art model-based algorithm Dreamer [29] in autonomous racing task with the LiDAR observation (LO) and occupancy reconstruction (OR). Here, we set them as two independent baselines Dreamer (LO) and Dreamer (OR), and train them by the process reward as suggested in the original paper. The winner of F1Tenth 2019 utilizes the Follow-the-Gap [47] (FTG) method, which is a classical path-planning algorithm and the expert policy for demonstration in RacingDreamer, so we consider it as a significant baseline in our experiments. Furthermore, we compare our method with pure model-free DRL methods and train five leading baselines including TRPO [40], PPO [41], DDPG [48], TD3 [49] and SAC [50]. According to our results and previous work [15], their performances are always unsatisfactory and we therefore record the best one of them as MFRL (best). Importantly, we also invite 5 professional players of Formula Student to control the agent by direction keys on a game console. Human players are allowed to practice on arbitrary tracks, and observe the scenarios from the bird's-eye view (BE) or the following view (FV).

Training Settings. At the beginning of an episode, the racing agent is placed randomly on the finish line and each episode has a maximum length of 5,000 time-steps. We train our ResRace for only 2M time-steps to show its advantage on convergence speed. Dreamer and other model-free DRL baselines are trained for 5M time-steps. For fairness, all baselines utilize the same MLP with 2 layers of 256 neurons and are trained for 5 trials with independent random seeds. For evaluation, FTG and all learning-based methods are tested for

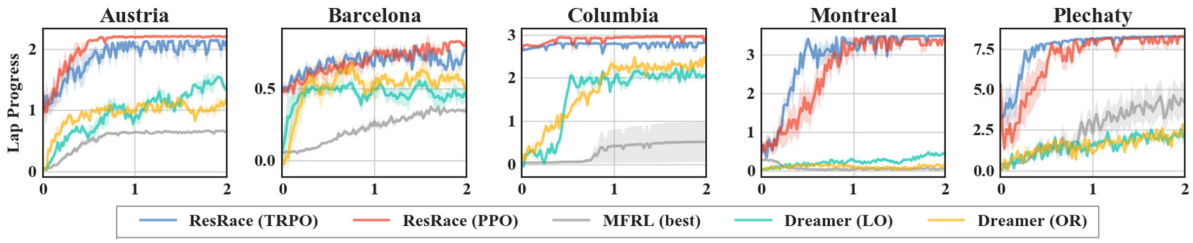

Fig. 4. The training processes of ResRace and other baselines in 2M time-steps. Each episode consists of 5,000 time-steps. ResRace and other baselines are trained for 5 trials with independent random seeds.

100K steps with 5,000 time-steps per episode. Human players are evaluated on the 5 tracks for 50,000 time-steps with 2 views and the average performance of their 5 best episodes is calculated as the baseline Human (BE) and Human (FV).

Performance Metrics. We first visualize the training curves in 2M steps as Fig. 4 and analyze their learning ability and property. Besides, we divide the lap progress by the episode length to calculate the surrogate lap time for all baselines according to their validation performance. Importantly, when a method can not complete a full lap (except the longest track Barcelona), we consider it fails on this track and its lap time is recorded as *Dnf.* (do not finish). Moreover, we evaluate their real-time performance on the model size and maximum frame rates in the simulator with the same scenario and hardware for 100K test time-steps.

Robustness Validation. To evaluate the robustness of ResRace, we trial our models in different scenarios without retraining. We first increase the potential force F from $0.7F$ to $1.3F$ to validate its sensitivity to MAPF intensity. Meanwhile, we increase the original friction factor $\mu = 0.80$ from 0.55 to 0.95 to validate our models under different dynamics. Besides, we conduct the cross-validation by testing the trained agent on unseen tracks for 100K time-steps. We use the ratio of test results and results of the model trained on the test track to evaluate the generalization ability.

Ablation Studies. We test the performance of MAPF controller and pure PPO independently to illustrate their complementary property. Furthermore, to emphasize the contribution of modification in APF, we remove the anti-stagnation probability γ and replace the balance vector $\mathcal{X}_{xy}$ with $[0.05, 0.05]^T$ to restore the imbalance force. Besides, we replace our reward design with the original setting in the simulator as $\mathcal{R} = c\Delta p - \mathcal{P}_s$, where $c = 100$ is a constant scalar and Δp denotes the lap progress in one time-step.

VI. Experimental Results

In this section, we evaluate the baselines and visualize their experimental results. We also further discuss the causes of obtained results according to the simulator demonstration. In addition, we provide the racing process of human players and ResRace agent on the Barcelona as a supplement to facilitate our behavior analysis.

Training Process. In Fig. 4, due to the guidance of the MAPF controller, the lap progress of ResRace is much greater than other baselines in the initial episodes. Model-free baselines and Dreamer agents explore with random actions so they collide with the track edges in most cases. Compared with MFRL (best), the MAPF-based policy effectively reduces the difficulty of policy optimization and highly contributes to performance improvement. Besides, though Dreamer reaches comparable performances of our method in 2M steps on three tracks, its prediction process and extensive modeling lead to much higher algorithm complexity and more time consumption with the same episodes. Meanwhile, on the tracks with changing widths like Montreal and Plechaty, it is difficult for conventional model-free methods and Dreamer to learn the practicable policy.

TABLE I
Lap Time Comparison

Methods	Lap Time (s)↓				
	Aus.	**Bar.**	**Col.**	**Mon.**	**Ple.**
FTG [47]	41.92	123.34	34.03	*Dnf.*	11.98
MFRL (best)	*Dnf.*	151.52	35.82	*Dnf.*	13.12
Dreamer (LO) [15]	37.56	102.21	**27.60**	*Dnf.*	16.34
Dreamer (OR) [15]	37.68	100.23	28.12	*Dnf.*	16.42
Human (BE)	37.57	97.89	29.24	25.54	10.52
Human (FV)	**36.77**	**95.82**	28.05	**23.02**	10.35
ResRace (TRPO)	**36.71**	99.21	28.15	**23.74**	**10.06**
ResRace (PPO)	36.87	**95.79**	**27.59**	23.88	**9.84**

Dnf. = Do not finish

Performance Comparison. In Table I, the performance of ResRace is better than other methods in most cases though they are trained for more steps. Especially, other approaches fail to learn the practicable policy to pass the sharp bend on the Montreal. Meanwhile, though Dreamer takes FTG as expert demonstration and outperforms the model-free methods, it tends to learn the sub-optimal policy and is still hard to defeat human players. For human players, the following view provides more intuitive observation so it is easier to control the distance from the track edges. As a result, the performance of Human (FV) is better than that of Human (BE). Besides, in most cases, ResRace can reach the comparable performance of human players with the following view. Although ResRace takes more $+0.62s$ than humans on the Montreal, the invited players always fail at bends. Our results and demonstration indicate that ResRace adopts an aggressive policy and controls the vehicle at the dynamics limitation like humans.

Efficiency. Although we harmonize the network with the same MLPs, the number of required models are various for different methods. Experimental results show our method

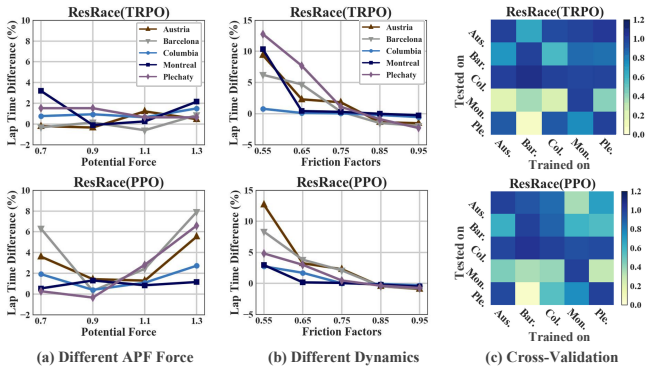

Fig. 5. The results of robustness and cross-validation. (a) and (b) show the performance of ResRace with different potential forces and dynamics, respectively. The initial fraction factor $\mu = 0.8$. (c) illustrates the results of ResRace tested on unseen tracks. Each block represents the progress ratio of the tested model and the trained model. The darker the block, the better the generalization performance of the tested model.

requires two networks for value estimation and policy generation with only 0.48M parameters. While Dreamer (LO) and Dreamer (OR) require 3.52M and 2.31M parameters for additional system dynamics modeling. The off-line methods like SAC and TD3 require an additional policy network and possess 0.72M parameters. Meanwhile, ResRace with TRPO and PPO reach 211.27Hz and 210.43Hz sampling rates in the simulator, which are 9.7% and 10.3% lower than pure TRPO and PPO, respectively. These results illustrate that the MAPF controller only slightly affects the real-time performance of our paralleled framework. However, due to the latent imagination mechanism, Dreamer's sampling rates are respectively 38.62Hz and 30.51Hz with LO and OR, which requires much more computational resources to achieve the same real-time performance.

Change Potential Force. Fig. 5(a) demonstrates that ResRace is not sensitive to the potential field intensity and indicates the parameters can be easily adjusted in the range of $\pm 30\%$. We notice that the performance of ResRace (PPO) with $1.3F$ is reduced by about 8% on Barcelona. According to the simulation, when potential field intensity is too weak or strong, the racing agent yaws and navigates along the wave line due to the tiny or excessive transverse force. These undesired problems directly cause the collision at bends especially on the narrow track. Meanwhile, ResRace (TRPO) demonstrates lower parameter sensitivity than ResRace (PPO).

Change Tire Friction. The results in Fig. 5(b) shows our method can adapt to the changing tire friction. With the increased friction, the racing agent can reduce its lap time with higher acceleration and turning speed. While the friction is reduced, though the performance shrinks slightly, it still finishes the laps stably. For instance, the bends are smoother in Barcelona than those in others so the agent requires less friction to pass them. On the contrary, when driving at the sharp bends of Austria and Barcelona, the racing agent must pass them with a longer path or lower velocity to reduce the required friction, which is consistent with experiences of human drivers.

Cross-Validation. As shown in Fig. 5(c), trained models maintain their performance when tested on the track with similar features. For example, the model trained on Austria can handle the sharp bends in Plechaty. On the contrary, the irregular and coarse edges of Montreal challenge the generalization of models trained on the other track, so that the test results on Montreal are always unsatisfactory. Meanwhile, the model trained on complex tracks demonstrates outstanding reliability. Specially, on Columbia, the model trained on Barcelona outperforms the Columbia baseline model and reaches $27.55s$ per lap, which beats the professional player by $0.5s$ and proves that the model trained on the challenging tracks process the outstanding generalization ability.

TABLE II
THE RESULTS OF ABLATION EXPERIMENTS

Methods	Lap Progress ↑				
	Aus.	Bar.	Col.	Mon.	Ple.
MAPF Controller	1.31	0.43	2.64	0.64	6.25
Pure PPO	0.69	0.52	0.11	0.75	6.13
w/o Modification	2.15	0.81	2.72	**3.61**	7.59
w/o Reward Design	2.14	0.71	2.65	3.24	8.08
ResRace (PPO)	**2.26**	**0.87**	**3.01**	3.49	**8.47**

Ablation Studies. We validate contributions of crucial components in ResRace (PPO) as shown in Table II and ResRace (TRPO) has similar performances. As the ablation results, the MAPF controller is workable in most cases and its maximum progress matches the initial performance of ResRace in Fig. 4. Meanwhile, TRPO and PPO can hardly work independently on these tracks. These results demonstrate that the initial exploration of ResRace is dominated by MAPF policy, and the subsequent improvement is attributed to the DRL module. For the modification in APF, experimental results (w/o Modification) illustrate that it contributes to the performance of our framework, especially on Columbia and Plechaty. It improves the performance by $+10.1\%$ on the tracks except for Montreal. When the track is wide, the modification decreases the lateral potential force excessively and causes slow steering. Thus, the performance without reward design outperforms ResRace slightly. Besides, we replace our reward design with the original process reward in the simulator. According to the results (w/o Reward Design), our reward design can improve the average performance by 7.61%. In the simulation, the action penalty can effectively restrain the redundant steering and deceleration. Concurrently, with the discrete reward signal $\mathcal{R}_{fin}$, the agent learns to exit the last bend rapidly and sprint to the finish line. These results prove that our reward design can describe the task finely and effectively reduce the lap time. While after removing the reward design, the performance of ResRace is still better than most of the baselines.

Behavior Analysis. As shown in Fig. 6, we visualize the trajectories and velocity of the ResRace agent and human players to analyze their behaviors. On the long straight, the behaviors of our agent are consistent with that of human players. However, as marked by red boxes, ResRace agent selects more aggressive routes at the bends. Obviously, our agent keeps close to the track edge to reduce speed loss during

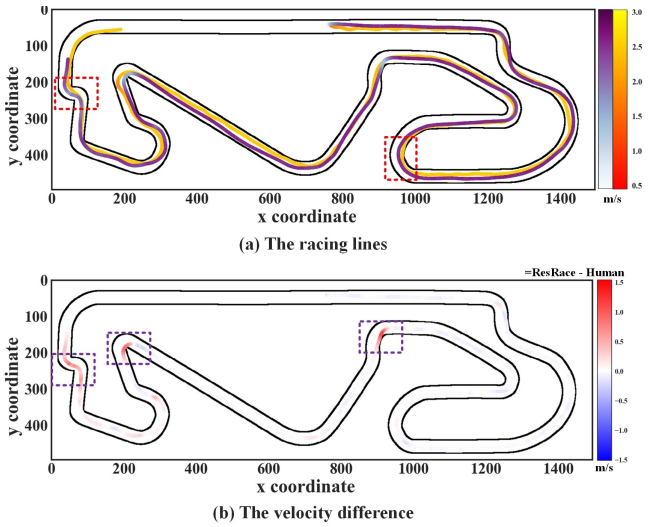

Fig. 6. The behavior demonstration on Barcelona. (a) is the racing lines of human player (purple) and ResRace (PPO) (orange). (b) is the velocity difference between the human player and ResRace (PPO). The red curves in purple boxes show ResRace (PPO) passes the bends faster than the invited players.

turning while the players prefer to decelerate to avoid the collision. Hence, though the selected actions are similar at the smooth bends, their speed difference is obvious at the sharp and continuous bends. As the purple boxes shown in Fig. 6(b), our agent chooses a later braking point and larger turning radius during passing the sharp bends, which bring the higher entering and exiting speed. In summary, the above results indicate that the behaviors of our agent are similar to that of human players in most cases while our agent learns to drive with higher speed and more aggressive racing line so our framework achieves better performance. From our driving experiences, the engine should maintain the maximum output power on the straight. However, our agent occasionally releases the throttle, resulting in a slightly inferior straight performance to human players.

VII. Conclusion

In this paper, we propose a residual policy learning method ResRace to solve the problems of the existing methods in exploration efficiency, robustness and prior information dependence. We analyze the complementarity between the DRL module and MAPF controller. Under the guidance of the MAPF policy, the residual policy can be easily optimized. Therefore, we use only MLP with few parameters as the policy network, which effectively accelerates the training process and ensures real-time performance. The experimental results demonstrate that our model-free framework can outperform a series of leading approaches and reach comparable level of human players. Meanwhile, ResRace works robustly with different policy otimization methods, APF hyperparameters and vehicle dynamics. The code and supplementary materials are available at *https://github.com/ispc-lab/resrace*.

We also notice the limitation of our method. Although the MAPF controller can constrain the risk exploration, the policy restriction is soft and may be covered by the DRL policy. Meanwhile, the performance of ResRace is significantly related to that of the fundamental policy. Although our residual learning framework only needs a workable policy to guide the exploration, this policy may still need fine tune. Additionally, our method still needs extensive validation in the real world. In future work, we will further improve the performance of our method and explore its application in more scenarios.

Acknowledgement

We acknowledge anonymous reviewers of IEEE Robotics and Automation Letters for providing valuable feedback of this work. We also thanks Jiayi Guan for helpful discussions. This work is funded by National Natural Science Foundation of China (No.61906138), and is supported by by Shanghai Municipal Science and Technology Major Project (No.2018SHZDZX01), ZJ Lab, and Shanghai Center for Brain Science and Brain-Inspired Technology, and the Shanghai Rising Star Program (No.21QC1400900).

References

[1] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, "Aggressive driving with model predictive path integral control," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1433–1440, 2016.

[2] A. Bulsara, A. Raman, S. Kamarajugadda, M. Schmid, and V. N. Krovi, "Obstacle avoidance using model predictive control: An implementation and validation study using scaled vehicles," tech. rep., SAE Technical Paper, 2020.

[3] U. Rosolia and F. Borrelli, "Learning how to autonomously race a car: A predictive control approach," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 6, pp. 2713–2719, 2020.

[4] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," in *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 305–313, 1989.

[5] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, *et al.*, "End to end learning for self-driving cars," *arXiv preprint arXiv:1604.07316*, 2016.

[6] Y. Pan, C.-A. Cheng, K. Saigol, K. Lee, X. Yan, E. Theodorou, and B. Boots, "Agile autonomous driving using end-to-end deep imitation learning," *arXiv preprint arXiv:1709.07174*, 2017.

[7] Y. E. Wang, G.-Y. Wei, and D. Brooks, "Benchmarking tpu, gpu, and cpu platforms for deep learning," *arXiv preprint arXiv:1907.10701*, 2019.

[8] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[9] J. Hwangbo, I. Sa, R. Siegwart, and M. Hutter, "Control of a quadrotor with reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 2, no. 4, pp. 2096–2103, 2017.

[10] W. Koch, R. Mancuso, R. West, and A. Bestavros, "Reinforcement learning for uav attitude control," *ACM Transactions on Cyber-Physical Systems*, vol. 3, no. 2, pp. 1–21, 2019.

[11] C. You, J. Lu, D. Filev, and P. Tsiotras, "Advanced planning for autonomous vehicles using reinforcement learning and deep inverse reinforcement learning," *Robotics and Autonomous Systems*, vol. 114, pp. 1–18, 2019.

[12] C. Yang, K. Yuan, Q. Zhu, W. Yu, and Z. Li, "Multi-expert learning of adaptive legged locomotion," *Science Robotics*, vol. 5, no. 49, 2020.

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

[14] M. Jaritz, R. De Charette, M. Toromanoff, E. Perot, and F. Nashashibi, "End-to-end race driving with deep reinforcement learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2070–2075, IEEE, 2018.

[15] A. Brunnbauer, L. Berducci, A. Brandstätter, M. Lechner, R. Hasani, D. Rus, and R. Grosu, "Latent imagination facilitates zero-shot transfer in autonomous racing," *arXiv preprint arXiv:2103.04909*, 2021.

[16] 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.

[17] A. Kendall, J. Hawke, D. Janz, P. Mazur, D. Reda, J.-M. Allen, V.-D. Lam, A. Bewley, and A. Shah, "Learning to drive in a day," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 8248–8254, 2019.

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

[19] M. F. Elmorshedy, W. Xu, F. F. El-Sousy, M. R. Islam, and A. A. Ahmed, "Recent achievements in model predictive control techniques for industrial motor: A comprehensive state-of-the-art," *IEEE Access*, vol. 9, pp. 58170–58191, 2021.

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

[21] A. Jain, M. O'Kelly, P. Chaudhari, and M. Morari, "BayesRace: Learning to race autonomously using prior experience," in *Proceedings of the 4th Conference on Robot Learning (CoRL)*, 2020.

[22] G. Williams, N. Wagener, B. Goldfain, P. Drews, J. M. Rehg, B. Boots, and E. A. Theodorou, "Information theoretic mpc for model-based reinforcement learning," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1714–1721, IEEE, 2017.

[23] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, "Information-theoretic model predictive control: Theory and applications to autonomous driving," *IEEE Transactions on Robotics*, vol. 34, no. 6, pp. 1603–1622, 2018.

[24] G. Bellegarda and K. Byl, "An online training method for augmenting mpc with deep reinforcement learning," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5453–5459, IEEE, 2020.

[25] A. Mehta, A. Subramanian, and A. Subramanian, "Learning end-to-end autonomous driving using guided auxiliary supervision," in *Proceedings of the 11th Indian Conference on Computer Vision, Graphics and Image Processing*, pp. 1–8, 2018.

[26] T. Weiss and M. Behl, "Deepracing: Parameterized trajectories for autonomous racing," *arXiv preprint arXiv:2005.05178*, 2020.

[27] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[28] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1334–1373, 2016.

[29] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi, "Dream to control: Learning behaviors by latent imagination," in *International Conference on Learning Representations (ICLR)*, 2019.

[30] A. Plaat, W. Kosters, and M. Preuss, "Model-based deep reinforcement learning for high-dimensional problems, a survey," *arXiv preprint arXiv:2008.05598*, 2020.

[31] A. Brunnbauer and L. Berducci, "Racecar Gym," *Github Repository:* $github.com/axelbr/racecar\_gym$, 2020.

[32] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation in robotics, games and machine learning," 2017.

[33] C. W. Warren, "Global path planning using artificial potential fields," in *1989 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 316–317, IEEE Computer Society, 1989.

[34] F. Bounini, D. Gingras, H. Pollart, and D. Gruyer, "Modified artificial potential field method for online path planning applications," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, pp. 180–185, 2017.

[35] C. Y. Tazibt, N. Achir, P. Muhlethaler, and T. Djamah, "Uav-based data gathering using an artificial potential fields approach," in *2018 IEEE 88th Vehicular Technology Conference*, pp. 1–5, IEEE, 2018.

[36] A. Singletary, K. Klingebiel, J. Bourne, A. Browning, P. Tokumaru, and A. Ames, "Comparative analysis of control barrier functions and artificial potential fields for obstacle avoidance," *arXiv preprint arXiv:2010.09819*, 2020.

[37] J. Sun, J. Tang, and S. Lao, "Collision avoidance for cooperative uavs with optimized artificial potential field algorithm," *IEEE Access*, vol. 5, pp. 18382–18390, 2017.

[38] J. Garcia and F. Fernández, "Safe exploration of state and action spaces in reinforcement learning," *Journal of Artificial Intelligence Research*, vol. 45, pp. 515–564, 2012.

[39] T. Johannink, S. Bahl, A. Nair, J. Luo, A. Kumar, M. Loskyll, J. A. Ojea, E. Solowjow, and S. Levine, "Residual reinforcement learning for robot control," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 6023–6029, IEEE, 2019.

[40] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International Conference on Machine Learning (ICML)*, pp. 1889–1897, PMLR, 2015.

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

[42] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," *arXiv preprint arXiv:1506.02438*, 2015.

[43] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine learning*, vol. 8, no. 3, pp. 229–256, 1992.

[44] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International Conference on Machine Learning (ICML)*, pp. 1928–1937, PMLR, 2016.

[45] J. P. LaSalle, "The 'bang-bang' principle," *IFAC Proceedings Volumes*, vol. 1, no. 1, pp. 503–507, 1960.

[46] T. Seyde, I. Gilitschenski, W. Schwarting, B. Stellato, M. Riedmiller, M. Wulfmeier, and D. Rus, "Is bang-bang control all you need? solving continuous control with bernoulli policies," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 34, 2021.

[47] 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.

[48] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas, "Dueling network architectures for deep reinforcement learning," in *International Conference on Machine Learning (ICML)*, pp. 1995–2003, PMLR, 2016.

[49] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International Conference on Machine Learning (ICML)*, pp. 1587–1596, PMLR, 2018.

[50] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *2018 International Conference on Machine Learning (ICML)*, pp. 1861–1870, PMLR, 2018.