

Article

Safe Reinforcement Learning for Competitive Autonomous Racing: Integrated State–Action Mapping and Exploration Guidance Framework

Yuanda Wang ^{1,*} , Jingyu Liu ¹ , Xin Yuan ^{1,2}  and Jiacheng Yang ¹ 

¹ School of Automation, Southeast University, Nanjing 210096, China; jingyu.liu@seu.edu.cn (J.L.); xinyuan@seu.edu.cn (X.Y.); 220221953@seu.edu.cn (J.Y.)

² School of Computer Science and Engineering, Nanyang Technological University, Singapore 639798, Singapore

* Correspondence: wangyd@seu.edu.cn

Abstract

Autonomous race driving has emerged as a challenging domain for reinforcement learning (RL) applications, requiring high-speed control while adhering to strict safety constraints. Existing RL-based racing methods often struggle to balance performance and safety, with limited adaptability in dynamic racing scenarios with multiple opponent vehicles. The high-dimensional state space and strict safety constraints pose significant challenges for efficient learning. To address these challenges, this paper proposes an integrated RL framework that combines three novel components: (1) a state mapping mechanism that dynamically transforms raw track observations into a consistent representation space; (2) an action mapping technique that rigorously enforces physical traction constraints; and (3) a safe exploration guidance method that combines conservative controllers with RL policies, significantly reducing off-track incidents during training. Extensive experiments conducted in our simulation environment with four test tracks demonstrate the effectiveness of our approach. In time trial scenarios, our method improves lap times by 12–26% and increases the training completion rate from 33.1% to 78.7%. In competitive racing, it achieves a 46–51% higher average speed compared to baseline methods. These results validate the framework's ability to achieve both high performance and safety in autonomous racing tasks.

Keywords: autonomous driving; reinforcement learning; safety constraint; action mapping; state mapping



Academic Editor: Yashar Mousavi

Received: 19 April 2025

Revised: 13 June 2025

Accepted: 23 June 2025

Published: 24 June 2025

Citation: Wang, Y.; Liu, J.; Yuan, X.; Yang, J. Safe Reinforcement Learning for Competitive Autonomous Racing: Integrated State–Action Mapping and Exploration Guidance Framework. *Actuators* **2025**, *14*, 315. <https://doi.org/10.3390/act14070315>

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Autonomous race driving represents a specialized yet critical subset of general autonomous driving tasks, distinguished by its extreme performance requirements, including precision control at high speeds and deliberate operation at vehicle dynamics limits. While these requirements differ markedly from conventional urban driving scenarios that prioritize passenger comfort and regulatory compliance, the advanced control strategies developed for racing, such as maneuvering at the limits of tire–road friction and split-second overtaking decisions, directly inform safety-critical applications in general autonomous driving, particularly in extreme scenarios like emergency obstacle evasion. This dual relevance, combining both theoretical challenges in control systems and practical applications in vehicle safety, has made autonomous racing an active area of research in academia and industry [1–3].

The objective of autonomous racing is to complete laps as fast as possible. It requires the vehicle to operate at high speeds while maintaining safety and stability. To address this challenging driving task, numerous advanced control strategies have been proposed for autonomous race driving, including classical control [4–8], learning-based intelligent control [9,10], and so on. Among these strategies, classical control methods basically focus on known path tracking. For example, Viadero-Monasterio et al. [11] proposed an event-triggered LPV path tracking controller that improves safety, comfort, and communication efficiency under network delays. In contrast, reinforcement learning (RL) [12] offers distinct advantages by automatically learning optimal policies through environmental interaction while handling high-dimensional states and continuous actions. This makes RL particularly suitable for mastering the extreme control challenges in autonomous race driving, ranging from precise friction-limit handling to competitive overtaking maneuvers.

Early RL-based approaches focused on establishing baseline capabilities. Jaritz et al. [13] pioneered an end-to-end RL framework that directly mapped sensory inputs to control outputs, establishing foundational techniques for high-speed racing scenarios. Building upon this direct mapping approach, Hell et al. [14] later developed a specialized LiDAR-only observation method with strong generalization capabilities across unfamiliar tracks, advancing sensor-specific implementations for practical racing applications. As the field progressed, researchers developed increasingly sophisticated techniques to refine control performance. Zhang et al. [15] introduced a residual policy learning framework that enhanced baseline controllers through iterative experience, improving adaptability to dynamic racing conditions. This advancement in policy refinement contributed to the broader goal of achieving superhuman performance, which was later demonstrated by Fuchs et al. [16], whose RL agent successfully surpassed human expert lap times in simulation environments. In addition, Evans et al. [17] conducted comprehensive comparisons of various RL architectures for controlling autonomous racing vehicles, providing valuable insights into effective algorithm selection for different racing scenarios.

While high performance is the primary goal in autonomous racing, ensuring safety during both training and deployment presents a critical challenge. To address safety challenges in autonomous racing, researchers have developed many approaches focusing on different aspects of the problem. At the fundamental level of vehicle handling limits, Wang et al. [18] introduced an action mapping (AM) mechanism combined with RL that enables vehicles to safely operate at friction boundaries while maximizing handling capability, effectively preventing instability during high-speed maneuvers. In parallel, Gottschalk et al. [19] proposed a hybrid and hierarchical approach that integrates classical optimal control for system dynamics with RL for collision avoidance, leveraging surrogate models to accelerate training and achieve real-time control for dynamic overtaking in autonomous racing. Building upon basic vehicle control safety, researchers have addressed the challenge of dynamic uncertainties in racing environments. Niu et al. [20] leveraged adversarial learning within an RL framework to bridge the dynamics gap, achieving remarkable safety by eliminating violations during training. Complementing this approach, Liu et al. [21] enhanced policy robustness through a Sequential Actor–Critic architecture that leverages historical trajectory information, significantly improving learning efficiency and stability during high-speed maneuvers. Chen et al. [22] integrated Hamilton–Jacobi (HJ) reachability theory into RL-based control algorithms to provide enhanced safety guarantees while maintaining competitive racing speeds. Extending this approach, Stachowicz et al. [23] developed risk-sensitive control methods that explicitly quantify and balance performance objectives against safety constraints in high-speed racing scenarios.

The above methods are basically deployed on simulation platforms [24,25]. To address the practical challenges of real-world implementation, Cai et al. [26] developed

an innovative approach combining imitation learning (IL) with RL for vision-based autonomous racing. Their method effectively overcomes the limitations of traditional approaches that are either computationally intensive or sensitive to environmental variations, while also addressing the sample efficiency and safety trade-offs inherent in pure IL or RL implementations.

As autonomous control of single vehicles has achieved significant progress, research focus has shifted toward competitive racing scenarios that require complex multi-car interactions. A curriculum-based RL method that systematically increases task difficulty during training is proposed in [27]. This approach strategically introduces competitive elements only after basic driving skills are mastered, enabling more efficient learning of complex behaviors. To address the collision avoidance issue in competitive scenarios, Yuan et al. [28] established an initial hierarchical collision avoidance framework that effectively reduced collisions but encountered scalability limitations as opponent numbers increased. Addressing these limitations, Thakkar et al. [29] advanced the field with an end-to-end RL architecture incorporating attention mechanisms to dynamically prioritize relevant opponent vehicles, significantly improving performance in densely contested racing environments. Notably, Wurman et al. [30] developed Gran Turismo Sophy, a deep RL agent that achieved championship-level performance in a commercial racing simulator, demonstrating both high speed and tactical behaviors such as overtaking and blocking. Beyond algorithmic advances, observation space design remains a critical challenge in competitive racing environments. The approach in [31] incorporates sensor data from vehicle-mounted LiDARs or cameras, resulting in high-dimensional observation spaces that can challenge RL algorithm learning efficiency, particularly in complex racing scenarios with multiple vehicles. These high-dimensional representations provide rich environmental information but require substantial computational resources and may introduce latency in real-time control systems. In contrast, approaches in [27–29] utilize lower-dimensional observation spaces primarily consisting of vehicle pose, motion states, and limited track data from the car's forward view. While computationally more efficient, these approaches often inadequately represent opponent vehicles within the observation space, limiting their effectiveness in competitive racing environments.

In summary, despite the significant progress made in the above works, the current literature still faces the following common challenges:

- Limited adaptability to dynamic and competitive scenarios: Many classical and RL-based methods are primarily validated in simplified or single-vehicle settings, lacking robustness in highly dynamic racing scenarios with multiple opponent vehicles.
- Safety–performance trade-off: While some methods provide safety guarantees, they may sacrifice racing performance, or vice versa. Achieving both high safety and high performance in aggressive racing remains challenging.
- Insufficient handling of variable and high-dimensional observations: Existing approaches often struggle with efficiently processing variable-sized or high-dimensional sensory data, especially when the number and position of opponents change rapidly.

These challenges motivate the integrated framework proposed in this paper, which aims to enhance adaptability, observation processing, safety, and sample efficiency in competitive autonomous racing.

In this paper, we focus on the challenge of developing high-speed autonomous racing under dynamic and competitive scenarios, where safety constraints and opponent interactions must be simultaneously addressed. The key difficulties include (1) maintaining vehicle stability near the traction limits during high-speed maneuvers; (2) efficiently processing variable-dimensional observations when opponents enter/exit the field of view; (3) ensuring safe exploration during RL training to avoid catastrophic failures. To address

these challenges, we propose an integrated RL framework that combines state mapping, action mapping, and safe exploration guidance to achieve both performance and safety. The three key innovations of this work are summarized as follows:

1. A state mapping mechanism that dynamically transforms raw track observations into a consistent representation space, enabling seamless policy transfer between different racing scenarios.
2. An action mapping technique that rigorously enforces physical traction constraints while maximizing the vehicle's handling capabilities.
3. A safe exploration guidance method that combines conservative controllers with RL policies, significantly reducing off-track incidents during training.

The remainder of this paper is structured as follows: Section 2 presents the vehicle dynamics model that forms the foundation of our racing environment. Section 3 formulates the autonomous racing task as a Markov Decision Process (MDP). Sections 4–6, respectively, elaborate on three key methodological contributions: the state mapping approach for dynamic environment representation (Section 4), the action mapping mechanism for traction constraint enforcement (Section 5), and the safe exploration guidance framework (Section 6). Section 7 details the implementation of the proposed RL framework. Comprehensive experimental results, including both time trial and competitive racing scenarios across multiple track configurations, are presented and analyzed in Section 8. Finally, Section 9 concludes the paper and outlines directions for future research.

2. Vehicle Model

To characterize the vehicle's kinematic and dynamic behavior, we employ the single-track vehicle model, which has been extensively validated in vehicle handling research [32,33]. As depicted in Figure 1, this simplified representation consolidates the front and rear wheels into single virtual wheels positioned at each axle's centerline. The complete nomenclature is detailed in Table 1. The model incorporates several simplifying assumptions: (1) negligible longitudinal and lateral load transfer effects; (2) uniform road–tire friction characteristics; and (3) planar motion on a flat surface. Under these assumptions, the vehicle dynamics naturally decouple into distinct longitudinal (acceleration/deceleration) and lateral (steering) components for analytical convenience.

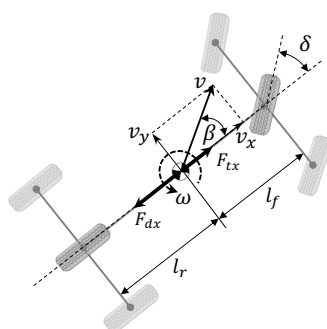


Figure 1. Single-track vehicle model.

Table 1. Nomenclature of vehicle model.

Symbol	Description (Unit)	Symbol	Description (Unit)
δ	Steering angle (rad)	β	Sideslip angle (rad)
v_x	Longitudinal velocity (m/s)	v_y	Lateral velocity (m/s)
v	Resultant velocity (m/s)	ω	Yaw rate (rad/s)
l_f	Distance from CG to front axle (m)	l_r	Distance from CG to rear axle (m)
F_{tx}	Longitudinal tire force (N)	F_{dx}	Longitudinal frictional drag force (N)

The longitudinal model of the vehicle can be derived from the analysis of longitudinal forces:

$$m\dot{v}_x = F_{tx} - F_{dx}, \quad (1)$$

where m is the total mass of the vehicle, F_{tx} is the longitudinal tire force, and F_{dx} is the longitudinal frictional resistance. The longitudinal tire force can take two forms: driving force and braking force. In this work, we consider a vehicle driven by an electric motor, whose torque output characteristics are divided into two regions within the range from zero speed to maximum speed: the constant torque region and the constant power region. When the motor speed n_m is less than the base speed n_b of the motor, the motor operates in the constant torque region, where its output torque is proportional to the motor control signal and can deliver the maximum torque. When the motor speed n_m exceeds the base speed n_b , the motor operates in the constant power region, and the output torque is limited by the maximum power $P_{\max}$, preventing it from maintaining the maximum torque. Combining these two cases, the driving force F_{mx} of the vehicle is expressed as

$$F_{mx} = \begin{cases} \frac{K_m u_m}{R_w} & n_m < n_b, \\ \min \left[\frac{K_m u_m}{R_w}, \frac{P_{\max}}{n_m R_w} \right] & n_m \geq n_b, \end{cases} \quad (2)$$

where K_m is the torque coefficient of the motor, R_w is the radius of the wheel, and $u_m \in [0, 1]$ is the normalized motor control signal. The braking force F_{bx} is generated by the vehicle's braking system and acts on all wheels. Assuming the braking force is proportional to the braking control signal, we have

$$F_{bx} = K_b u_b, \quad (3)$$

where K_b is the braking force coefficient and $u_b \in [0, 1]$ is the braking control signal. Since in normal conditions, acceleration and braking signals should not be input simultaneously, these two signals can be combined into one single longitudinal control signal $u_x \in [-1, 1]$. $u_x = 1$ indicates maximum acceleration command, and $u_x = -1$ indicates maximum braking force command.

The longitudinal frictional resistance F_{dx} is opposite to the vehicle's direction of motion and is mainly composed of two parts: the aerodynamic drag F_{aero} and the tire rolling resistance F_{roll} . The equivalent aerodynamic drag of the vehicle in a windless environment is expressed as

$$F_{aero} = \frac{1}{2} \rho_A C_d A_f v_x, \quad (4)$$

where ρ_A is the air density, C_d is the vehicle's aerodynamic drag coefficient, and A_f is the vehicle's frontal area. The vehicle's tire rolling resistance is approximately proportional to the vertical load on the tires, which is expressed as

$$F_{roll} = f_r m g, \quad (5)$$

where f_r is the rolling resistance coefficient and g is the acceleration due to gravity.

The lateral and rotational dynamics of the vehicle can be described by establishing force equilibrium along the lateral direction and moment equilibrium about the vertical axis, expressed as

$$m\dot{v}_y = F_{yf} + F_{yr} \quad (6)$$

$$I_z \dot{\omega} = F_{yf} l_f - F_{yr} l_r \quad (7)$$

where I_z is the vehicle's yaw moment of inertia; F_{yf} and F_{yr} represent the lateral tire forces at the front and rear axles, respectively. The tire forces are modeled using the Magic Formula tire model, with a detailed derivation available in [34].

The front wheel steering angle δ is driven by the electric power-assisted steering system, with the input being the normalized angular velocity control signal $u_y \in [-1, 1]$. Here, $u_y = 1$ or $u_y = -1$ indicates that the power-assisted steering system drives the front wheels to rotate with the maximum angular velocity $\dot{\delta}_{\max}$, and $u_y = 0$ indicates that the current front wheel angle is maintained.

The vehicle's longitudinal and lateral control forces depend on the friction between the tires and the road surface, also known as traction. During high-speed race driving, significant braking and steering inputs can cause the vehicle to exceed the traction limit, leading to uncontrollable states such as sliding or oversteering. In practice, the upper limit of this friction force depends on various physical factors, including road surface material, tire size, tire pressure, tire temperature, etc. Assuming these physical factors remain constant, the friction circle model can be used to describe the upper limit constraint of the tire's friction force. The friction circle model shows that the total force applied to the vehicle cannot exceed the maximum friction force, that is,

$$|\vec{F}_x + \vec{F}_y| = |\vec{F}_{xy}| < \mu_{\max} F_z, \quad (8)$$

where $\vec{F}_x$ and $\vec{F}_y$ are the longitudinal and lateral forces acting on the vehicle, respectively, and F_z is the total vertical load on the tire. Ignoring aerodynamic lift and downforce, $F_z = mg$. $\mu_{\max}$ is the maximum tire–road friction coefficient. This constraint can also be described as $|\vec{a}_{xy}| < \mu_{\max} g$, where $\vec{a}_{xy}$ is the vehicle's total acceleration in the horizontal direction, which can be measured by onboard accelerometers. To ensure that the race car remains stable and controllable during competitive driving while maximizing the vehicle's handling, the driving strategy should aim to stay as close as possible to this limit without exceeding it.

3. MDP Model for Racing Driving

The basic idea of RL is to iteratively optimize the control strategy of an agent based on the input–output experience from the agent's interactions with the environment, with the goal of maximizing the cumulative reward at each step. In order to apply RL to the racing driving task, it is necessary to first establish an appropriate MDP model to represent the interaction between the autonomous driving agent and the car–track environment. An MDP is typically defined by a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, $\mathcal{P}$ is the state transition function, $\mathcal{R}$ is the reward function, and γ is the discount factor. In the autonomous racing driving scenario, the state of the autonomous driving agent at time step t is $s_t \in \mathcal{S}$, which selects and executes the control action a_t according to its control policy π . The agent then transitions to a new state s_{t+1} based on the state transition function $\mathcal{P}$ and receives a reward $r_{t+1} = \mathcal{R}(s_t, a_t)$ based on the reward function. The state space, action space, and reward functions for the racing control MDP constructed in this paper are defined as follows.

3.1. State Space

The state variables in the racing driving MDP are composed of three parts: The first set of state variables describes the vehicle's own measured motion state, including vehicle speed v , yaw rate ω , and front wheel steering angle δ . The second set of state variables describes the vehicle's relative position on the track, as shown in Figure 2, including the lateral distance d_y from the vehicle's center to the track's centerline, and the angle ϕ between the vehicle's heading and the tangent direction of the track. The third set of state variables

reflects the positional information of the track edges and opponent vehicles ahead within a certain distance, composed of a series of observation vectors. V_{L_i} and V_{R_i} represent the vectors from the vehicle's center pointing toward the left and right track edges, respectively. V_{C_i} represents the observation vector from the vehicle's center to the opponent vehicles within the observable range. Specifically, taking the left edge observation vector as an example, assuming the center coordinates of the race car are (X_v, Y_v) and the coordinates of the left track edge point are (X_{L_i}, Y_{L_i}) , the observation vector is defined as

$$\vec{V}_{L_i} = \mathbf{R}(\psi)[X_{L_i} - X_v, Y_{L_i} - Y_v]^T, \quad (9)$$

where ψ is the vehicle's heading and $\mathbf{R}(\psi) \in \mathbb{R}^{2 \times 2}$ is the corresponding rotation matrix that transforms the vector from the inertial coordinate system to the vehicle's coordinate system.

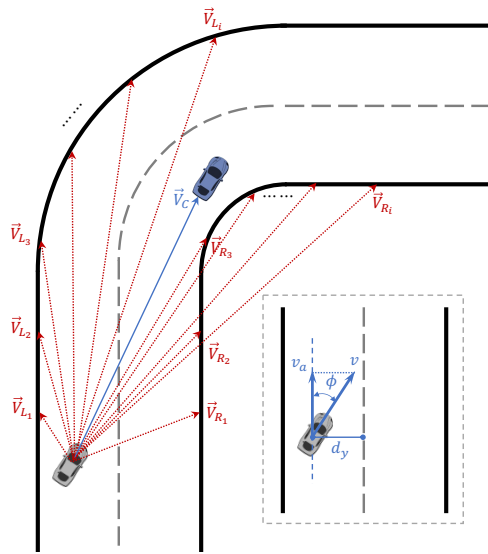


Figure 2. Part of state variables of racing driving.

By concatenating the two sets of N_E edge observation vectors from both the left and right sides, we obtain the edge observation feature: $\mathbf{V}_E \in \mathbb{R}^{4 \times N_E}$. By concatenating all the opponent vehicle observation vectors $\vec{V}_{C_i}$, we form the opponent observation feature $\mathbf{V}_C$. It is important to note that, in practice, there may be no opponent vehicles within the observable range ahead, or there may be one or more opponent vehicles. Therefore, the dimension of the vector $\vec{V}_{C_i}$ is uncertain, and the specific handling method is detailed in Section 4. Finally, by combining the three sets of state variables, we obtain the state vector of the racing driving MDP, which is

$$s_t = [v, \omega, \delta, d_y, \phi, \mathbf{V}_E, \mathbf{V}_C]. \quad (10)$$

3.2. Action Space

Based on the aforementioned vehicle model, the control signals consist of a combination of the normalized throttle/brake (longitudinal) control signal u_x and the normalized steering (lateral) control signal. Although in typical RL-based control methods, the actions are equivalent to the control signals, this is not the case in this study. Therefore, the corresponding lateral control action a_x and longitudinal control action a_y are additionally defined, and the control vector is represented as $a_t = [a_x, a_y]$. The specific relationship between the actions and real control signals is detailed in Section 5.

3.3. Reward Function

The goal of racing driving is to complete the track in the shortest possible time while avoiding dangerous situations such as running off the track, skidding, and collisions between cars. Therefore, a well-designed reward function is required to correctly represent the optimization objectives, and it is defined as

$$r_t = r_{\text{vel}} + r_{\text{out}} + r_{\text{skid}} + r_{\text{coli}} + r_{\text{slow}} + r_{\text{ww}}. \quad (11)$$

where $r_{\text{vel}} = v_a = v \cos \phi$ (shown in Figure 2) is the speed reward, which represents the projection of the vehicle's speed along the track tangent. This term serves as an effective proxy for minimizing lap time. This approach provides more frequent reward signals during continuous operation compared to sparse lap time rewards. It also encourages both high straightaway speeds and optimal cornering speeds that maintain momentum through turns.

Except for the speed reward, all other components in the reward function are implemented as binary penalties. Each penalty term takes a value of -100 when the corresponding undesired condition is triggered and 0 otherwise. The off-track penalty r_{out} is applied when the center of the vehicle crosses the track boundary. The skidding penalty r_{skid} is incurred when the vehicle's lateral acceleration exceeds the friction limit, as defined in Section 2. The collision penalty is triggered when the distance between the ego vehicle and an opponent vehicle falls below a safety threshold, defined as 1.2 times the vehicle length. Additionally, the slow driving penalty r_{slow} and wrong-way driving penalty r_{ww} are included to discourage unnecessarily slow speeds (below 20 km/h) and driving in the wrong direction, respectively.

It should be noted that although the proposed safety mechanisms can avoid skidding and car collisions, corresponding penalty terms are still included for the purpose of training baseline comparison methods.

4. State Mapping for Track Observations

Based on the description of the vehicle's state space mentioned above, the observation of the track ahead consists of a series of vectors indicating the positions of the track edges and opponent cars. Although this method of describing the state is common in RL-based autonomous driving research, it lacks flexibility in dealing with the varying number of opponent vehicles ahead. Specifically, when there are no opponent vehicles or multiple opponent vehicles within the observable range, it becomes difficult to accurately express the current state without changing the state space dimension. Additionally, the mismatch in the state space dimension caused by incorporating the information of opponent vehicles ahead creates inconvenience in transferring driving strategies between the two common racing scenarios: with and without opponent vehicles. For example, a driving strategy trained in a scenario without opponent vehicles cannot be applied to a scenario with opponent vehicles and must be retrained, which prevents the use of techniques like transfer learning to fully utilize the results of previous training to improve training efficiency.

To enhance the flexibility of the racing driving strategy in the above situations, we propose the state mapping method for the track environment's forward observation. The specific approach is shown in Figure 3. If there are opponent vehicles within the observable range ahead, the overtaking feasible area is dynamically defined based on the positions of all observable opponent vehicles, and the original track edge observation vectors are mapped to the edge of the overtaking feasible area. To construct the feasible area, the first step is to determine whether the feasible area is on the left or right side of the opponent vehicle. The method is as follows: when the front vehicle is driving in the middle of the

track, with space for safe overtaking on both sides, the side closer to the race car is chosen as the feasible area to avoid unnecessary detouring; when the front vehicle is close to one side of the track edge, the larger space on the opposite side is selected as the feasible area to ensure safety.

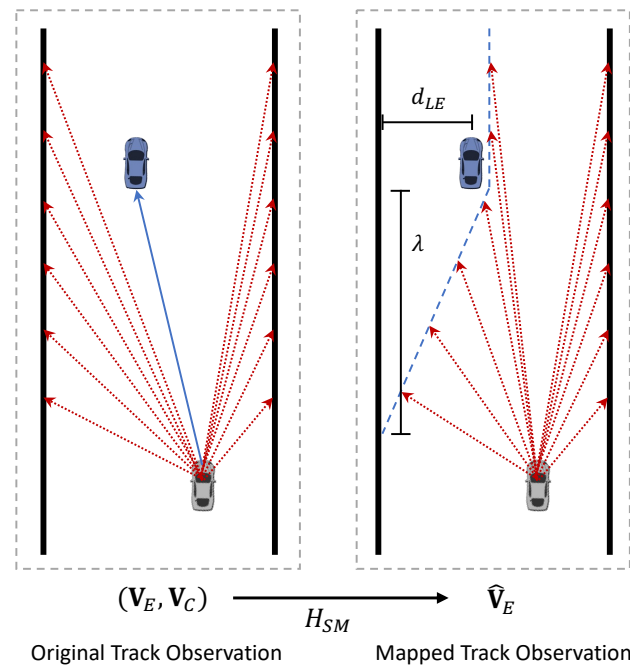


Figure 3. State mapping for race car's observation of the track ahead.

Next, based on the position of the front vehicle on the track and its relative position to the current car, the observation vectors of the original track edges are mapped. The focus is on calculating the positional deviation between the original track edge point and the feasible area edge point. Taking the right-side overtaking case shown in Figure 3 as an example, let point L_i be located on the left edge of the track within the observable range. Let the distance along the track's tangent direction from this point to the opponent vehicle be d_{L_i} , where $d_{L_i} > 0$ indicates that the track edge point is ahead of the opponent vehicle and the opposite indicates it is behind. By moving point L_i a distance of T_{L_i} to the right, perpendicular to the track's tangent direction, it lands on the left edge of the feasible area. This point is denoted as the left feasible area edge point $\hat{L}_i$, and the moved distance is defined as

$$T_{L_i} = \begin{cases} d_{LE} + \varepsilon & d_{L_i} \geq 0, \\ (1 + d_{L_i}/\lambda)(d_{LE} + \varepsilon) & -\lambda < d_{L_i} < 0, \end{cases} \quad (12)$$

where d_{LE} represents the distance between the opponent vehicle and the left edge of the track, ε is the lateral safety redundancy distance of the feasible area, and $\lambda > 0$ is the backward extension distance of the feasible area. When $-d_{L_i} > \lambda$, it indicates that the entire width of the current track is a feasible area. λ reflects the longitudinal safety redundancy of the overtaking feasible area. Similarly, when the right side of the opponent vehicle is considered as the feasible area, the track's right edge point R_i is moved left by a distance T_{R_i} to obtain the right feasible area edge point $\hat{R}_i$.

After obtaining the edge information of the feasible area, the original track edge observation vectors $\vec{V}_{L_i}$ or $\vec{V}_{R_i}$ can be mapped to the edge of the feasible area, forming a new track edge observation feature. This mapping relationship is expressed as

$$\hat{\mathbf{V}}_E = H_{SM}(\mathbf{V}_E, \mathbf{V}_C). \quad (13)$$

In addition, when the race car is driving in the feasible area processed by the state mapping, its observation of its relative position within the track will also change. This is equivalent to one side of the track narrowing while the car's lateral position remains unchanged. Therefore, the state variable d_y , which describes the distance from the vehicle to the track centerline, is also transformed into $\hat{d}_y$. After the track state space mapping processing, the state vector for the racing control MDP is represented as

$$\hat{s}_t = [v, \omega, \delta, \hat{d}_y, \phi, \hat{\mathbf{V}}_E]. \quad (14)$$

The proposed state mapping method dynamically constructs overtaking feasible areas by integrating the positional information of all observable leading opponents. Compared to the original vehicle state variables, this approach maintains a consistent state dimensionality regardless of the number of opponents. Although Figure 3 presents a single-opponent scenario for illustrative clarity, the method naturally generalizes to multi-opponent situations by uniformly processing all relevant vehicles within the observable range.

5. Action Mapping for Traction Constraint

The racing vehicle model provides traction constraints to ensure safe handling. Therefore, the control strategy of the racing car should strictly adhere to these constraints to avoid the car entering a dangerous state. This work introduces the action mapping mechanism from our previous work on RL control problems with constraints to handle these constraints. The basic principle of the action mapping mechanism is to establish a mapping between the unconstrained virtual policy π_V and the constrained real policy π_R (i.e., the actual controller), which transforms the original control problem with input constraints into a general saturated constraint control problem and optimizes it using RL algorithms. This method does not alter the original RL algorithm framework but ensures that the real policy corresponding to the virtual policy satisfies the constraint conditions through a specially designed mapping function. The key to this mechanism lies in constructing a one-to-one mapping function from the virtual policy space to the real policy space. In the following, the basic theory of the action mapping mechanism and the specific method for handling traction constraints are provided.

Consider the deterministic policy case in RL, where the virtual policy for action mapping is defined as

$$a = \pi_V(s), \quad s \in \mathcal{S}, a \in \mathcal{A}, \quad (15)$$

where the continuous state space $\mathcal{S}$ is a compact subspace of $\mathbb{R}^{N_s}$ and N_s is the dimensionality of the state. Since deterministic policies for continuous action spaces are often represented by neural networks and use nonlinear activation functions like tanh to constrain the output range, the action space $\mathcal{A}$ is defined as $[-1, 1]^{N_a}$, a compact subspace of $\mathbb{R}^{N_a}$, where N_a is the dimensionality of the action. The constrained true control policy is defined as

$$u = \pi_R(s), \quad s \in \mathcal{S}, u \in F(s), \quad (16)$$

where F is a set-valued map from $\mathcal{S}$ to the power set $P(\mathbb{R}^{N_a})$, which describes the constraint control input space $F(s)$ corresponding to any $s \in \mathcal{S}$. The set-valued map F can be described by its graph, $\text{Graph}(F) := \{(x, y) | x \in \mathcal{S}, y \in F(x)\}$. The graph is a subset of the product space $\mathbb{R}^{N_s} \times \mathbb{R}^{N_a}$, and it is the constrained state-action space for this MDP.

Let $\mathcal{H}$ be the set of all continuous functions that map from $\mathcal{S}$ to $\mathcal{A}$. Let $\mathcal{G}$ denote the set of all continuous functions $\pi : \mathcal{S} \rightarrow \bigcup_{s \in \mathcal{S}} F(s)$ that satisfy the condition that for any given $s \in \mathcal{S}$, $\pi(s) \in F(s)$. Specifically, $\mathcal{H}$ is the set of all virtual policies, and $\mathcal{G}$ is the set of all real policies.

According to the action mapping theorem (Theorem 1 in [35]), there exists a mapping $T : \mathcal{H} \rightarrow \mathcal{G}$ between the virtual policy $\pi_V \in \mathcal{H}$ and the real policy $\pi_R \in \mathcal{G}$, if and only if there exists a continuous mapping $H_{AM} : \text{Graph}(F) \rightarrow \mathcal{A}$ such that for any given $s \in \mathcal{S}$, the mapping $h_s : F(s) \rightarrow \mathcal{A}$ is a homeomorphism between $F(s)$ and $\mathcal{A}$. The mapping function h_s is defined as $h_s(a) = H_{AM}(s, a)$. The key to the existence of the policy mapping T is to find a mapping function H_{AM} , referred to as the action mapping function. The additional conditions and theoretical proofs that guarantee the existence of this mapping are also provided in reference [35].

Next, the action mapping mechanism is used to specifically handle the traction constraint for racing driving. As mentioned in Section 3.2, the actions and the control signals are not equivalent, which is different from typical RL-based control approaches, and this is to align with the application of the action mapping mechanism. Specifically, the virtual policy is defined as $a_t = \pi_V(s_t)$, where $a_t = [a_x, a_y]$ is the virtual action vector and $a_x, a_y \in [-1, 1]$ represent the virtual longitudinal and lateral control actions, respectively. The actual controller is represented as $u_t = \pi_R(s_t)$, where $u_t = [u_x, u_y]$ and u_x, u_y are acceleration/brake and steering signals, which have been defined in Section 2.

The constrained real policy can be obtained through the policy mapping T , which is specifically implemented through its corresponding action mapping function:

$$u_t = H_{AM}(\tilde{s}_t, a_t), \quad (17)$$

where $\tilde{s}_t = [v, \delta]$ is a subset of the state variables that only includes the state variables relevant to the traction constraints.

Due to the complexity of vehicle dynamics and the constraints, it is extremely difficult to provide an analytical expression for the mapping function. Therefore, we introduce a numerical approximation method from our previous work [18] to construct the mapping function. The basic idea of this approximation method is to adjust the action vector with respect to a boundary function $B(\tilde{s})$. The boundary function $B(\tilde{s})$ is the implemented version of $F(s)$ in this racing driving case. It indicates the safety boundary of the traction limits for the control inputs applied to the vehicle in state $\tilde{s}$. To determine the boundary function, we employ the single-track vehicle model established in Section 2 to iteratively adjust the states of v and δ at specified numerical intervals. At each iteration, we incrementally apply larger control inputs and verify whether these inputs adhere to the traction constraints.

However, the computational complexity of the full vehicle model, particularly due to the intricate tire force calculations, makes real-time computation of the boundary function prohibitively slow. While this approach is feasible for offline simulations, real-world applications require online boundary function computation to account for dynamic variations in vehicle performance and tire conditions. To address this challenge, we maintain the original longitudinal dynamic model while adopting a simplified kinematic model for lateral motion. This simplification assumes zero slip angles for both front and rear tires, with the vehicle's sideslip angle and yaw rate expressed as

$$\beta = \arctan\left(\frac{l_r \tan \delta}{l_f + l_r}\right), \quad (18)$$

$$\omega = \frac{v \tan \delta \cos \beta}{l_f + l_r}. \quad (19)$$

Although this simplification may slightly reduce the accuracy of traction limit estimation, we compensate by employing a conservative maximum friction coefficient, ensuring the resulting boundary function maintains robust safety margins.

The implementation of the action mapping from a_t to u_t at state s_t is carried out by projecting the action vector that exceeds the boundary $B(\tilde{s})$ onto the boundary itself, while preserving its original direction. For action vectors that lie within the boundary, no modification is made. Specifically, the action vector a_t is temporarily converted into polar coordinates $[\rho_a, \vartheta_a]$, where ρ_a and ϑ_a represent the vector length and direction, respectively. Using the boundary function $B(\tilde{s})$, we can obtain the maximum allowable length in a specific direction under state $\tilde{s}$, denoted as $\rho_{v,\delta,\vartheta_a}^{\max}$. Then, the implementation form of the action mapping function is represented by

$$u_t = H_{AM}(v, \delta, \rho_a, \vartheta_a) = \begin{cases} \mathcal{T}_{p \rightarrow c}[\rho_a, \vartheta_a] & \rho_a \leq \rho_{v,\delta,\vartheta_a}^{\max} \\ \mathcal{T}_{p \rightarrow c}[\rho_{v,\delta,\vartheta_a}^{\max}, \vartheta_a] & \text{otherwise,} \end{cases} \quad (20)$$

where $\mathcal{T}_{p \rightarrow c}$ is the operator for converting from polar coordinates to Cartesian coordinates. The details of the approach using a numerical method with the racing model to establish the mapping function are provided in reference [18].

Through the established action mapping function, unconstrained actions are mapped to a safe control signal. This method fundamentally prevents the race car from entering uncontrollable states, while fully utilizing the maximum tire–road friction, ensuring the safety of the autonomous racing driving strategy during both the training and execution phases.

6. Safe Exploration with a Guidance Policy

In addition to the traction constraint addressed above, another primary challenge is the risk of the car running off the track, especially during the initial exploration phase. Although the action mapping mechanism for traction constraint could keep the vehicle controllable, other inappropriate actions inside the safe boundary could also cause the car to deviate from the track.

To solve this issue, we further develop an action mapping-based safe exploration guidance method. A predefined conservative track-following controller (see Section 7 for details: Stanley lateral controller + PID speed regulator) is introduced as an initial guidance policy. Different from the conventional policy guided method for RL, in which the control actions from the guidance policy and the RL agent are directly merged, in this work, the action mapping mechanism is utilized, and it adaptively limits the exploration and updating boundary of the RL-based racing policy. The guidance policy is also iteratively updated during the training process.

Specifically, we define the guidance policy as $\pi_G : \mathcal{S} \rightarrow \mathcal{A}$ and the main policy as $\pi_M : \mathcal{S} \rightarrow \mathcal{A}$. To guarantee a stable updating process, the exploration range and the updating step of the policy is constrained with respect to the guidance policy. The update process of the main policy is defined as

$$\pi_M(s) \leftarrow \min_{a \in G(s)} \hat{Q}^{\pi_M}(s, a), \quad (21)$$

where $\hat{Q}^{\pi_M}(s, a)$ is the approximated state–action value function of the main policy. $G(s)$ is a set-valued map which maps from the state space $\mathcal{S}$ to the power set $P(\mathbb{R}^{N_a})$, which is

$$G(s) = \{a \mid \|a - \pi_G(s)\| \leq \mathcal{C}(s)\}, \quad (22)$$

where $\mathcal{C} : \mathbb{R}^{N_s} \rightarrow \mathbb{R}_+$ is the constraint function for safe exploration, which limits the maximum exploration distance of the main policy with respect to the guidance policy. The map is further represented as

$$G(s) = \{a + \pi_G(s) \mid a \in G'(s)\}, \quad (23)$$

where $G'(s)$ is also a set-valued map, defined as $G'(s) = \{a \mid \|a\| \leq \mathcal{C}(s)\}$. As mentioned before, the action space for the RL-based continuous control method is often defined as $[-1, 1]^{N_a}$. Hence, we define a set-valued map $L(x) = [-1, 1]^{N_a}$ to represent the unconstrained exploration range.

Similar to the action mapping framework for traction constraint, we use the graph of the set-valued map to explain the mapping approach. The graph of $G(s)$ is defined as $\text{Graph}(G) := \{(x, y) \mid x \in \mathcal{A}, y \in G(x)\}$. The graph of $L(s)$ is defined as $\text{Graph}(L) := \{(x, y) \mid x \in \mathcal{A}, y \in [-1, 1]^{N_a}\}$. $\text{Graph}(G)$ and $\text{Graph}(L)$ represent the real exploration space and the virtual exploration space. Similarly, we can also construct a mapping function that bridges the real and virtual exploration spaces with the help of $G'(s)$ and its graph $\text{Graph}(G')$. According to the action mapping theory, there exists a continuous mapping $H_{EM} : \text{Graph}(G') \rightarrow \mathcal{A}$, such that for any given $a \in \mathcal{A}$, the mapping $h_a : G'(s) \rightarrow \mathcal{A}$ is a homeomorphism. The mapping function is defined as $h_a = H_{EM}(s, a)$, and the implementation version of the mapping function is represented by

$$H_{EM}(s, a) = \frac{\sup_{\sigma \in \mathbb{R}} \{\sigma \mid \sigma a \in F'(s)\}}{\sup_{\sigma \in \mathbb{R}} \{\sigma \mid \sigma a \in \mathcal{A}\}} a, \quad (24)$$

where $H_{EM}(s, a) = 0$ when $a = 0$. Based on this mapping function H_{EM} , a one-to-one correspondence is established between the real exploration space $\text{Graph}(F)$ and the virtual exploration space $\text{Graph}(L)$ through $\text{Graph}(F')$. At this point, for any given state–action pair (s_t, a_t) in real exploration space, its corresponding exploration action τ_t in the exploration space is

$$\tau_t = H_{EM}(s_t, a_t - \pi_G(s_t)). \quad (25)$$

Conversely, given a state s_t and an exploration action τ_t , its corresponding action in the real exploration space can be calculated by

$$a_t = H_{EM}^{-1}(s_t, \tau_t) + \pi_G(s_t). \quad (26)$$

Note that $\text{Graph}(F')$ is determined exclusively by the constraint function $\mathcal{C}(s)$. Consequently, modifications to the guiding policy π_G do not influence the mapping function H_{EM} . This property allows us to leverage standard RL algorithms to iteratively optimize the main policy.

In addition, during training, as the main policy π_M is optimized, it may eventually surpass the performance of the guiding policy π_G . When this occurs, the current guidance policy can hinder further improvement of π_M . To address this, we evaluate the main policy every T_{eval} episodes, using lap time as the performance metric. If the difference in lap times between the two policies, denoted as $\Delta\pi$, exceeds a predefined threshold ε_t , we update the guiding policy by replacing it with the current main policy: $\pi_G \leftarrow \pi_M$.

7. Implementation with RL Algorithms

In this section, we detail the implementation of our proposed RL-based racing control approach, which integrates three key components: the state mapping module for track observations; the action mapping module for traction constraints; and the safe exploration guidance module. These proposed modules are compatible with most RL algorithms capable of handling continuous observation and action spaces. For illustration, we employ the classic Twin Delayed Deep Deterministic Policy Gradient (TD3) algorithm [36] to demonstrate the implementation process.

A block diagram of the proposed RL framework for autonomous racing is shown in Figure 4. The left part of the diagram illustrates the execution section of the framework, which demonstrates the direct interaction between the autonomous driving agent and the

racing environment. The state vector s_t is obtained from the environment observation. The edge observation feature $\mathbf{V}_E$ is defined as a set of 10 observation vectors per track edge. These vectors sample the track at increasing look-ahead distances ranging from 10 m to 100 m, with a uniform sampling interval of 10 m. The original state vector s_t is then processed by the state mapping module H_{SM} . The position information of the opponent vehicles $\mathbf{V}_C$ is integrated into the original track edge observation vector and forms $\hat{\mathbf{V}}_E$.

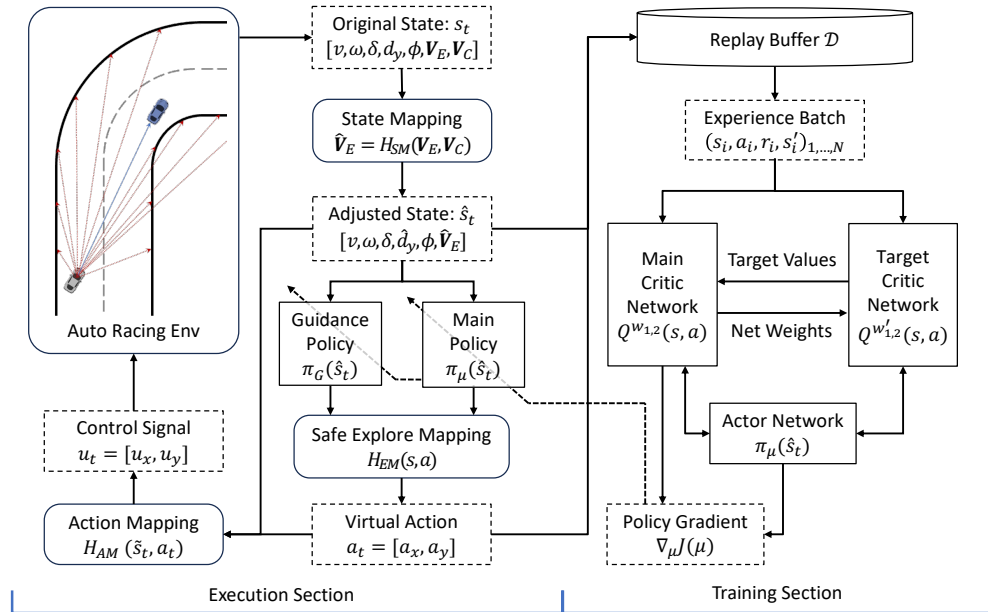


Figure 4. Structure of proposed RL framework for autonomous racing.

The processed state vector is subsequently input to the safe exploration guidance module. Within this module, an admissible control action is generated by synthesizing the guidance policy and the main policy, ensuring compliance with the predefined constraints on the exploration range. It should be noted that all state variables are normalized to either the interval $[-1, 1]$ or $[0, 1]$, based on their respective minimum and maximum observed values.

For the initial guidance policy, we implement a Stanley steering controller [37] for lateral control to track the centerline. Since the Stanley control law outputs a desired steering angle, while our model accepts steering rate command rather than a direct angle input, we introduce a secondary proportional controller to compute the required steering rate. The control law is given by

$$a_y = K_\delta \left(\phi - \arctan\left(\frac{K_{cte} \cdot d_y}{v + \zeta}\right) - \delta \right) \quad (27)$$

where K_δ is the steering rate gain; K_{cte} is the cross-track error gain; and ζ is a small constant introduced to avoid division by zero. For longitudinal control, we employ a PID controller to regulate speed, maintaining a constant reference velocity through acceleration and braking commands.

It is worth noting that the initial guidance policy is primarily designed to keep the vehicle on track during early exploration. As such, suboptimal yet safe parameter tuning is sufficient. During training, this policy is gradually replaced by the learned RL policy once the latter demonstrates superior performance, as discussed in Section 6.

Next, this action is fed into the action mapping module, where it is considered as the unconstrained virtual action. Then, by utilizing the action mapping function H_{AM} , the virtual action a_t is converted to a safe control signal u_t , which guarantees that the traction constraint is satisfied. After that, this control signal is used to control the racing car in the

environment. The racing environment then returns the reward based on the predefined reward function and the next state.

The right portion of Figure 4 depicts the RL training section of the proposed framework. We adopt the off-policy TD3 algorithm, motivated by preliminary experiments from our prior work on autonomous racing [18], where TD3 exhibited superior training stability compared to the on-policy PPO algorithm. In particular, TD3's twin-critic architecture effectively mitigates value overestimation while maintaining the stability required for safe operation near the limits of vehicle dynamics. Although our framework is compatible with various RL algorithms, the characteristics of TD3 make it especially well-suited for demonstrating the effectiveness of the proposed methods.

The actor and critic of the TD3 algorithm are implemented as fully connected neural networks. The actor network, denoted as $\pi_\mu(\hat{s})$ with trainable parameters μ , serves as the main policy in the safe exploration module in the execution section. The target network of the actor network shares the same structure, and it is denoted as $\pi'_\mu(\hat{s})$. Structurally, the actor network consists of two hidden layers, each comprising 256 neurons with ReLU activation. The output layer utilizes the tanh activation function to constrain actions within the range of $[-1, 1]$. To address the issue of value function overestimation, the TD3 algorithm employs a twin-critic architecture. The two critic networks, represented as $Q_{w_1}(s, a)$ and $Q_{w_2}(s, a)$ with parameters w_1 and w_2 , respectively, share an identical hidden layer structure with the actor network. While the hidden layers utilize ReLU activation functions, the output layers adopt linear activation to estimate Q-values without restriction. Corresponding target networks, denoted by $Q_{w'_1}(s, a)$ and $Q_{w'_2}(s, a)$, maintain the same architecture but employ slowly updated parameters to enhance learning stability. The updating rate for both target networks is set as 0.005.

The training process employs batch learning through an experience replay buffer $\mathcal{D}$, which stores state transition tuples $e_t = (\hat{s}_t, a_t, r_{t+1}, s_{t+1})$ collected during policy execution. The maximum size of the buffer is set as 1 million. Notably, the buffer records the adjusted state representation $\hat{s}_t$ rather than the original state. During each training iteration, a mini-batch of experience tuples $\{(s_i, a_i, r_i, s'_i)\}_{i=1}^N$ are randomly sampled from the replay buffer. The batch size $N = 256$ is empirically determined to balance learning stability and computational efficiency. The twin-critic networks are then independently optimized by minimizing the temporal difference (TD) error through the following loss function:

$$\mathcal{L}(w_j) = \frac{1}{N} \sum_{i=1}^N \left[y_i - Q_{w_j}(s_i, a_i) \right]^2, j \in \{1, 2\}, \quad (28)$$

where y_i is the target value from the target critic networks, that is,

$$y_i = r_i + \gamma \min_{j=1,2} Q_{w'_j}(s'_i, \pi_{\mu'}(s'_i)) + \epsilon. \quad (29)$$

where γ is the discount factor. To value overestimation in the target values, a clipped Gaussian noise is introduced: $\epsilon \sim \text{clip}(\mathcal{N}(0, \sigma_\epsilon^2), -c, c)$, where c denotes the clipping bound.

The actor network parameters μ are updated using a sampled approximation of the policy gradient, computed through the deterministic policy gradient theorem. Utilizing the same batch of transitions and the primary critic network $Q_{w_1}(s, a)$, the gradient estimate is given by

$$\nabla_\mu J(\mu) \approx \frac{1}{N} \sum_{i=1}^N \nabla_\mu \pi_\mu(s_i) \nabla_a Q_{w_1}(s_i, a_i)|_{a_i=\pi_\mu(s_i)}. \quad (30)$$

The training procedure of the race driving policy with the proposed framework is formally presented in Algorithm 1, where a Gaussian noise term $n_t \sim \mathcal{N}(0, \sigma_\mu^2)$ is

incorporated into the virtual action at each step to facilitate effective exploration. To enhance training stability, the algorithm implements a delayed policy update scheme whereby the actor network and target networks are only updated after every $T_{\text{delay}} = 2$ iterations of critic network optimization. Additionally, the training process includes periodic policy evaluation at predetermined intervals, enabling performance monitoring and preservation of optimal policy snapshots throughout the learning process.

Algorithm 1 Proposed RL Framework for Autonomous Race Driving

```

1: Initialize actor network parameters  $\mu$ , critic network parameters  $w_1, w_2$ 
2: Initialize target networks parameters  $\mu' \leftarrow \mu, w'_1 \leftarrow w_1, w'_2 \leftarrow w_2$ 
3: Initialize network update parameters  $\eta_a, \eta_c, \eta_{\text{tar}}$ 
4: Initialize replay buffer  $\mathcal{D} \leftarrow \emptyset$ 
5: Set initial guidance policy  $\pi_G$ 
6: Load race driving environment
7: for Episode = 1 to MaxEpisode do
8:   Initialize ego car and opponent cars (if applicable)
9:   Observe initial state  $s_1$ 
10:  Process with state mapping:  $\hat{s}_1 = H_{SM}(s_1)$ 
11:  for time step  $t = 1$  to MaxStep do
12:    Generate exploration action:  $\tau_t = \pi_\mu(\hat{s}_t) + n_t$ 
13:    Process with safe explore mapping:  $a_t = H_{EM}^{-1}(\hat{s}_t, \tau_t) + \pi_G(\hat{s}_t)$ 
14:    Process with action mapping:  $u_t = H_{AM}(\hat{s}_t, a_t)$ 
15:    Execute  $u_t$ , observe  $s_{t+1}$ , and receive  $r_{t+1}$ 
16:    Process with state mapping:  $\hat{s}_t = H_{SM}(s_t), \hat{s}_{t+1} = H_{SM}(s_{t+1})$ 
17:    Store  $(\hat{s}_t, a_t, r_{t+1}, \hat{s}_{t+1})$  in  $\mathcal{D}$ 
18:    Sample experiences  $\{(s_i, a_i, r_i, s'_i)\}_{i=1}^N \sim \mathcal{D}$ 
19:    Update critic networks  $w_j \leftarrow w_j - \eta_c \nabla_{w_j} \mathcal{L}(w_j)$  using Equation (28)
20:    if  $t \bmod T_{\text{delay}} = 0$  then
21:      Update actor network  $\mu \leftarrow \mu + \eta_a \nabla_\mu J(\mu)$  using Equation (30)
22:      Update critic and actor target networks:

$$w'_j \leftarrow \eta_{\text{tar}} w_j + (1 - \eta_{\text{tar}}) w'_j, \quad \mu' \leftarrow \eta_{\text{tar}} \mu + (1 - \eta_{\text{tar}}) \mu', \quad j = 1, 2$$

23:    end if
24:    if car violates safety constraints then
25:      break
26:    end if
27:    if Episode mod  $T_{\text{eval}} = 0$  then
28:      Run policy comparative evaluation of  $\pi_G$  and  $\pi_\mu$ 
29:      if  $\Delta_\pi > \varepsilon_t$  then
30:        Update guidance policy  $\pi_G \leftarrow \pi_\mu$ 
31:      end if
32:    end if
33:  end for
34: end for

```

8. Experiments and Results

This section presents a comprehensive evaluation of the proposed RL-based race driving approach within our developed racing simulation platform. In the following, we first detail the specifications of the simulation platform; then, we conduct experimental trials in two distinct racing scenarios—a time trial scenario without opponent vehicles and a competitive race scenario involving multiple opponents on the same track.

8.1. Simulation Environment

The racing driving simulation environment is built based on the single-track vehicle model established in Section 2, and the differential equations are numerically solved using the fourth-order Runge–Kutta (RK4) method. The simulation time step is 0.01 s. The vehicle model parameters used in the simulation are derived from a pure electric mid-size sedan, and its basic physical parameters are provided in Table 2.

Table 2. Physical parameters for racing car simulation.

Parameter	Value (Unit)	Parameter	Value (Unit)
Vehicle mass m	1860 (kg)	Front axle distance l_f	1.17 (m)
Rear axle distance l_r	1.77 (m)	Tire radius R_w	0.31 (m)
Maximum steering angle $\delta_{\max}$	35 (deg)	Rolling resistance coefficient f_r	0.015
Aerodynamic drag coefficient C_d	0.3	Air density ρ_A	1.225 (kg/m ³)
Frontal area A_f	2.05 (m ²)	Engine power $P_{\max}$	125 (kW)
Motor torque coefficient K_m	1550	Braking force coefficient K_b	16,422
Max tire–road friction $\mu_{\max}$	1.15	Gravitational acceleration g	9.81 (m/s ²)

To comprehensively and thoroughly test the performance of the driving strategy, four test tracks with different styles are built in the simulation environment. Figure 5 illustrates the track layouts, while Table 3 summarizes their basic information. In Figure 5, the starting line (finish line) and starting direction marked by red lines and white arrows in the respective track layout diagrams. Track A is a simple circular test track, Track B is a go-kart track with short straights and many curves, Track C is built to the same dimensions as the Ruisi Circuit located in Beijing, China, and Track D is built to the same dimensions as the Wanchi Circuit located in Nanjing, China.

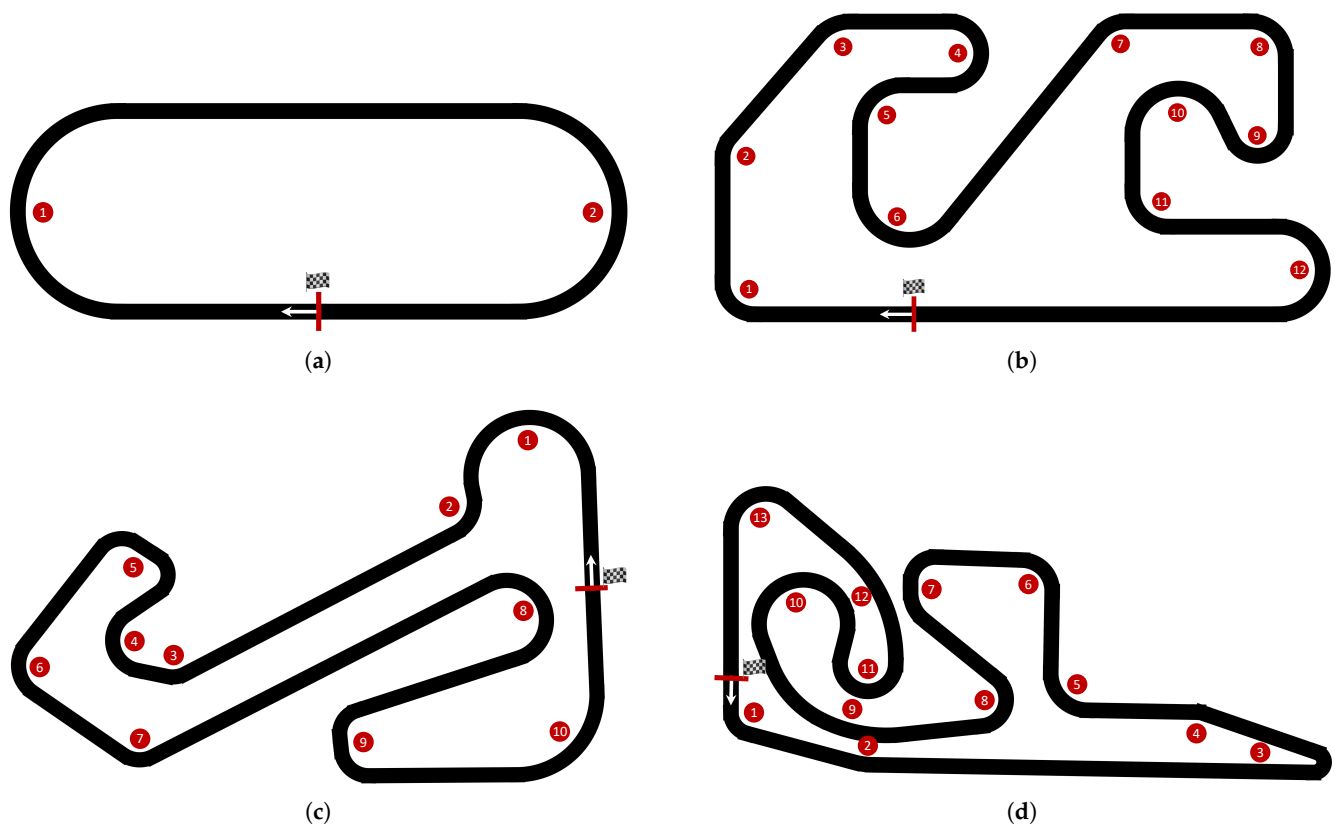


Figure 5. Layout of four test race tracks in simulation environment. The numbering of each track curve is marked by white digits on red circular based. (a) Layout of Track A. (b) Layout of Track B. (c) Layout of Track C. (d) Layout of Track D.

Table 3. Basic information for test race tracks.

Track Name	Length (m)	Width (m)	Number of Corners	Track Type
Track A	785	20	2	Oval
Track B	707	15	12	Karting
Track C	1408	10	10	Circuit
Track D	2001	12	13	Circuit

The simulation experiments were conducted on a hardware platform equipped with an Intel Core i9-13900K CPU, 32 GB RAM, and an NVIDIA GeForce RTX 4090 GPU. The software environment consists of Ubuntu 20.04 LTS as the operating system, with PyTorch 1.13 serving as the deep learning framework for neural network implementation and training.

8.2. Training and Results in Time Trial

The time trial scenario as a fundamental benchmark for evaluating autonomous racing performance. Since there is no opponent car in this setup, the assessment only focuses on the action mapping module and the safe exploration guidance module, while the state mapping module is not involved. To facilitate comparative analysis, we evaluate three RL-based driving policies: (1) TD3 is the standard implementation without additional modules. (2) TD3-AM extends TD3 by incorporating the action mapping (AM) modules to address the friction constraints. (3) TD3-AMGP combines a guidance policy (GP) for safe exploration guidance.

The training of three policies follows Algorithm 1, and the unused procedures are omitted from the algorithm. For each episode the car is initialized under the following conditions: (1) initial car position: random location on straight track segments; (2) initial velocity: $v_x \sim \mathcal{U}(0, 100)$ km/h; (3) other state variables: $\omega = \phi = \delta = 0$.

The assessment was conducted across four aforementioned track configurations. For each combination of policy and track, we conducted five independent training trials with distinct random seeds to ensure statistical robustness. Each trial consisted of 8000 episodes, with every episode spanning 5000 time steps (50 s). The accumulated reward values and termination reasons of each episode were recorded throughout training.

The learning progress comparison of three driving policies across four tracks is illustrated in Figure 6. The mean rewards across seeds are plotted as solid curves, and the corresponding standard deviations are represented by shaded regions. The learning curves show that both TD3-AM and TD3-AMGP converge after around 4000 training episodes (approximately 10 million training iterations). The average training time for each of these policies is approximately 5 h. Notably, the computational overhead introduced by the safe policy guidance mechanism in TD3-AMGP is negligible. For comparison, the standard TD3 implementation completes the same number of iterations in about 4 h, due to the absence of action mapping computations.

The learning curves analysis reveals distinct performance characteristics among the three driving policies in each track. On the straightforward Track A, all policies demonstrate comparable reward progression, indicating that the basic TD3 policy can achieve competent driving in simple environments without additional modules. However, the more complex Tracks B, C, and D clearly differentiate the modules' capabilities. The incorporation of action mapping (AM) in both TD3-AM and TD3-AMGP yields significant improvements over the baseline, with approximately 50–100% higher asymptotic rewards, by maintaining tire forces near the friction limits while preventing loss of control—enabling both faster lap times and greater stability. TD3-AMGP initially achieves higher rewards on complex tracks due to its pre-tuned guidance policy, which provides basic driving capability and prevents unsafe exploration behaviors like driving off the track. In contrast, TD3-AM requires more

time to learn viable and safe driving policies through exploration. The guidance policy updating technique of TD3-AMGP effectively avoids the driving policy from becoming trapped in the initial guidance policy, enabling TD3-AMGP to eventually converge to a performance level comparable to TD3-AM. As a result, both methods eventually converge to similar performance levels, with TD3-AMGP maintaining safer exploration throughout the training process.

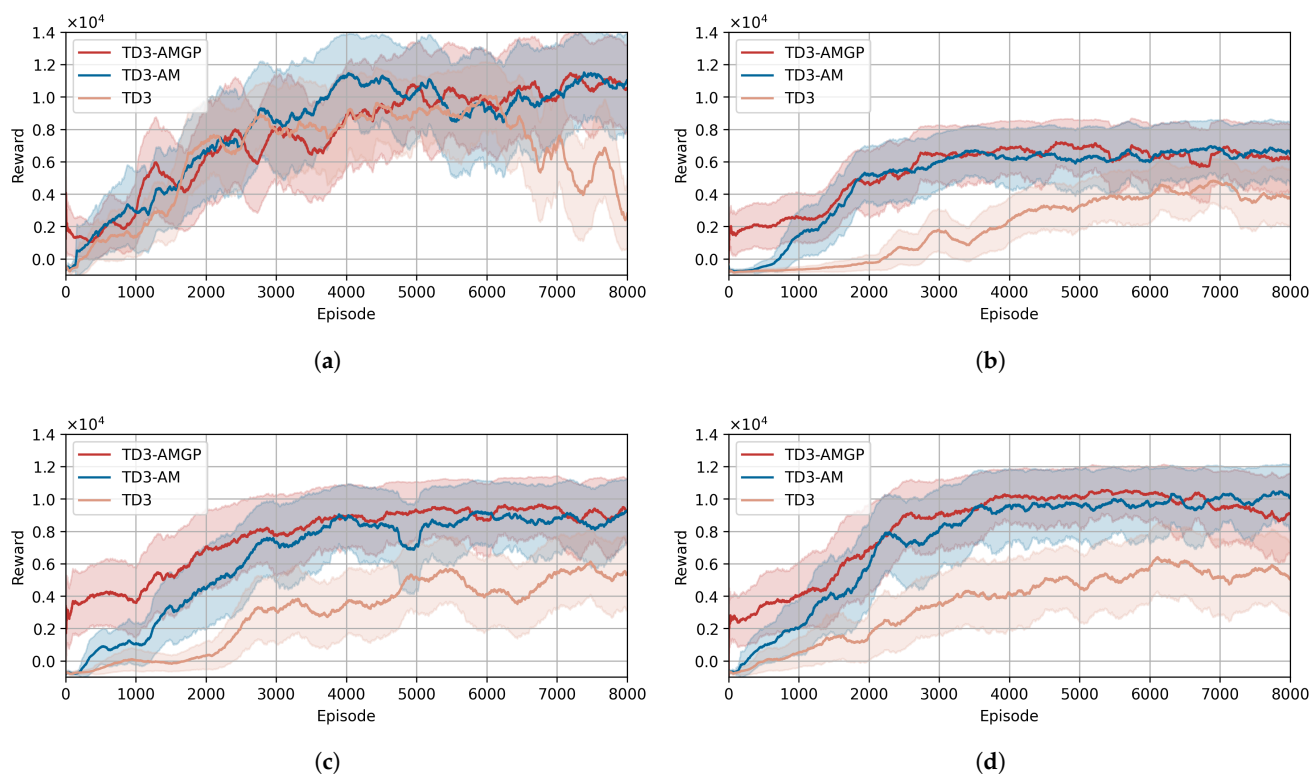


Figure 6. Learning curve comparison of three policies across four tracks in time trial scenario. (a) Learning curves in Track A. (b) Learning curves in Track B. (c) Learning curves in Track C. (d) Learning curves in Track D.

The episode termination reasons during policy training across four tracks are quantitatively analyzed. The results are visualized in Figure 7. The termination categories correspond to the definitions in the reward function: (1) ‘off track’ represents driving off the track; (2) ‘skid’ indicates exceeding tire friction limits; (3) ‘slow’ denotes termination due to excessively low driving speed; and (4) ‘WW’ signifies driving in the wrong direction.

The statistical results show generally similar patterns of termination reasons across the three driving policies on four tracks. Specifically, the TD3-AMGP policy, due to its incorporated guidance policy, demonstrates significantly fewer off-track terminations compared to the other two methods, with approximately 50% reduction relative to the baseline TD3 across all tracks. Furthermore, with the introduction of the action mapping module, both TD3-AM and TD3-AMGP completely avoid skid-induced terminations.

In addition, we compute the completion rate of all policies across the four tracks. The completion rate is defined as the proportion of training episodes that are successfully completed without triggering any termination conditions. The completion rates of TD3, TD3-AM, and TD3-AMGP are 33.1%, 58.7%, and 78.7%, respectively, with TD3-AMGP achieving the highest completion rate across all four tracks.

It should be noted that although TD3-AMGP implements a guidance policy, random exploratory actions are still added during training, which may occasionally cause off-track incidents. This design is intentional—without maintaining some off-track ex-

periences, the policy cannot effectively learn to prevent such situations. Therefore, the proposed safe exploration module can reduce but not entirely eliminate off-track occurrences during training.

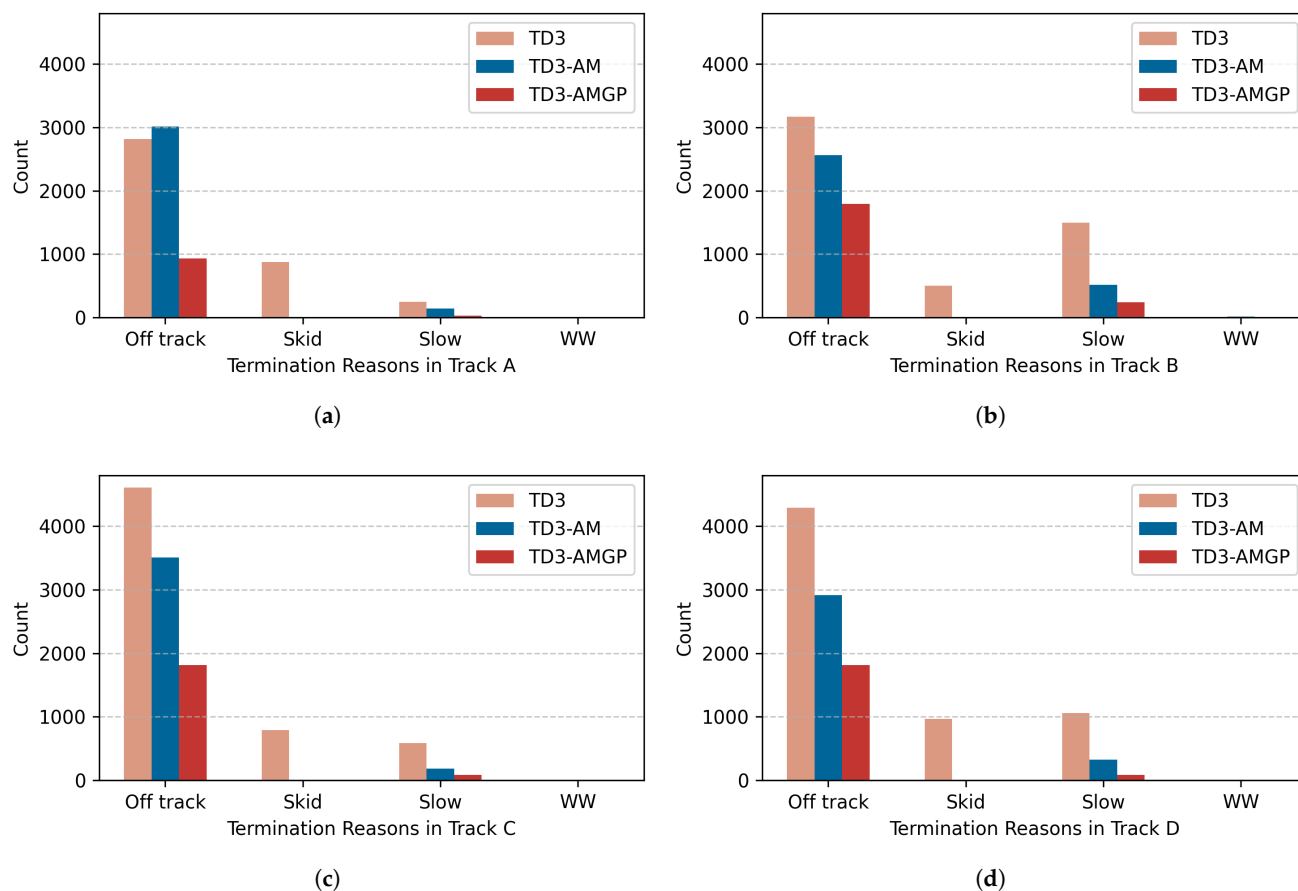


Figure 7. Count of termination reasons of 8000 training episodes of three driving policies across four tracks in time trial scenario. (a) Count of termination reasons in track A. (b) Count of termination reasons in track B. (c) Count of termination reasons in track C. (d) Count of termination reasons in track D.

To evaluate and demonstrate the driving skills of the trained driving policies, the driving policies are saved and tested in an evaluation episode for every 10,000 training iterations. The lap time of a ‘flying lap’ is used to quantify driving skills. The evaluation episodes begin with the vehicle centered on the finish line with motion states $\omega = \phi = \delta = v = 0$, and the exploration noise is removed. If the driving policy completes at least two laps without failure, the second lap time is recorded as the ‘flying lap’ metric.

The best lap time of all saved driving policies evaluated in four tracks is summarized in Table 4. The results demonstrate that both TD3-AM and TD3-AMGP policies, which incorporate the action mapping technique, achieve significantly faster lap times compared to the baseline. Specifically, the TD3-AMGP policy shows only marginal improvement over TD3-AM on Track B, while maintaining comparable performance on other tracks. This observation aligns with our theoretical expectation that the primary purpose of introducing the guidance policy (GP) is not to improve lap time performance but rather to improve safety during training. Consequently, as evidenced by the experimental data, TD3-AMGP achieves essentially equivalent maximum lap speeds to the TD3-AM approach while providing additional safety benefits during the learning process.

The optimal lap trajectories achieved by the TD3-AMGP policy across four test tracks are visualized in Figure 8, where the vehicle’s instantaneous speed is color-coded using

a jet colormap, with the starting line and driving direction annotated. The trajectory analysis reveals that the learned policy has mastered fundamental racing techniques, including (1) smooth racing lines following the professional ‘out-in-out’ cornering principle, (2) precise apex clipping maneuvers, and (3) optimized acceleration/deceleration strategies. Particularly impressive is the policy’s performance on the more challenging real-world tracks (Tracks C and D), where it demonstrates professional-level competence in handling complex corner sequences—maintaining ideal racing lines while executing perfect speed adjustments through consecutive turns.

Table 4. Best flying lap time comparison of three driving policies across four tracks.

	Track A	Track B	Track C	Track D
TD3	31.51 s	55.31 s	1 m 20.25 s	1 m 37.37 s
TD3-AM	28.64 s	43.31 s	1 m 04.70 s	1 m 25.57 s
TD3-AMGP	27.71 s	41.06 s	1 m 04.68 s	1 m 26.17 s

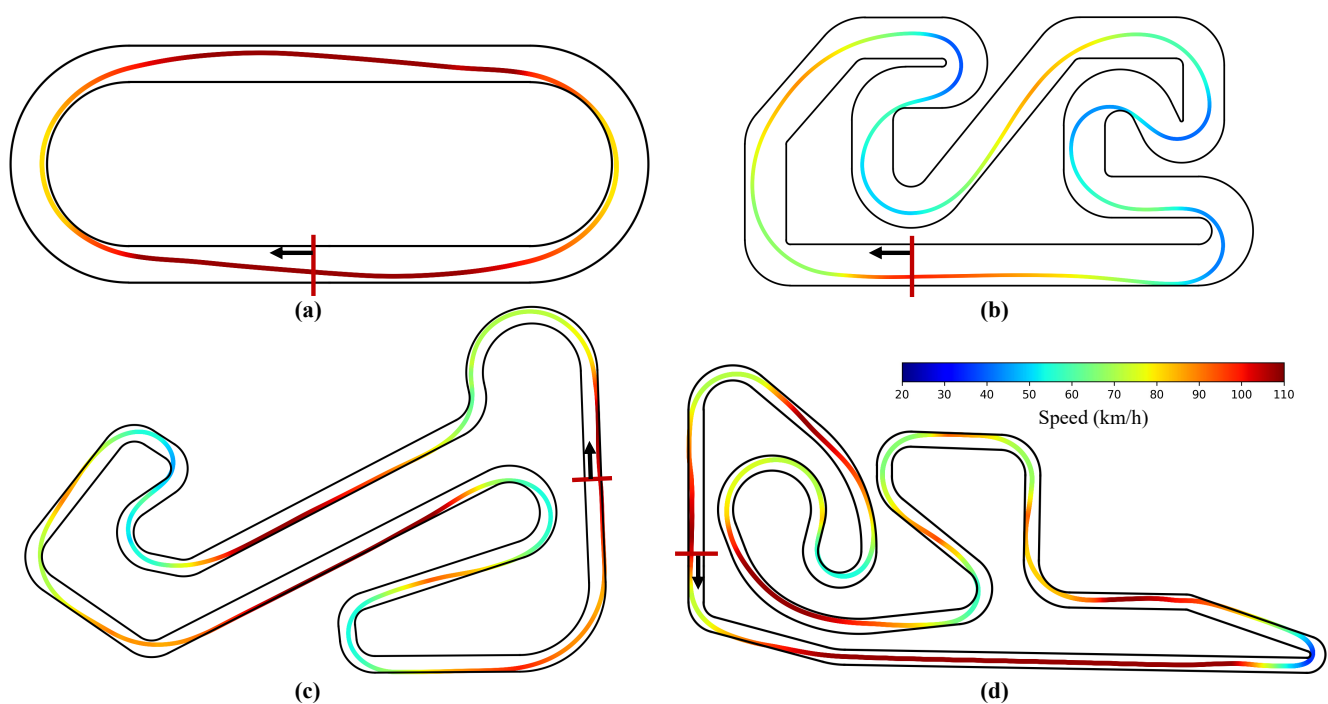


Figure 8. Best flying lap trajectories of TD3-AMGP driving policy in four tracks. The trajectories are color-coded by instantaneous speed (warmer hues indicating higher velocities). (a) Trajectory in Track A. (b) Trajectory in Track B. (c) Trajectory in Track C. (d) Trajectory in Track D.

8.3. Training and Results in Competitive Racing

Building upon our previous time trial experiments that evaluated autonomous racing fundamentals in single-vehicle scenarios, we now introduce dynamic opponent vehicles to assess the proposed state mapping technique. The comparative evaluation maintains consistency with prior methodology by examining three policy variants: (1) TD3-AM (baseline) is the same as the policy used in the time trial experiments. To maintain the same length of the opponent car observation vector $\mathbf{V}_C$, we only observe the nearest leading opponent car. (2) TD3-AMSM extends TD3-AM by incorporating the state mapping (SM) module to enhance the observation of opponent vehicles ahead. (3) TD3-AMSMGP integrates both SM and the guidance policy (GP) for safe exploration in competitive environments. The safe exploration guidance necessarily operates on the state-mapped representation when opponent vehicles are present. A standalone TD3-AMGP configuration would be incompatible since the guidance policy requires the transformed state variables ($\hat{d}_y, \hat{\mathbf{V}}_E$) that only the state mapping module can provide.

The training procedures for the three policies in the competitive race experiments maintain consistency with the time trial experiments, with the key distinction being the incorporation of opponent vehicles. Each opponent car is initialized with randomized parameters: a longitudinal separation of 50–150 m along the track axis, an initial velocity uniformly distributed between 20 and 60 km/h, and a variable lateral displacement from the track centerline. The opponent vehicles are controlled by predetermined controllers that maintain their initial lateral displacement while preserving the initialized velocity.

Notably, in the current setup, these opponent vehicles move at constant speeds, and their motion states remain constant over time. In addition, these opponent cars operate independently without any awareness of the ego car's motion, serving purely as dynamic environmental obstacles to evaluate the ego vehicle's overtaking capabilities.

Under these simplified assumptions, the feasible overtaking area is primarily determined by the opponent's position rather than dynamic behaviors such as speed changes. Accordingly, our state mapping module focuses on spatial configuration and does not explicitly encode opponent speed or acceleration. This design allows us to address variable-dimension observation challenges while preserving policy transferability and training efficiency. This experimental setup also ensures that the assessment focuses specifically on the ego vehicle's autonomous racing performance rather than modeling competitive interactions between vehicles. To account for the increased complexity of competitive racing compared to time trial scenarios, we have extended the training duration from 8000 to 10,000 episodes per trial to ensure sufficient policy convergence. The accumulated reward values and termination reasons of each episode were logged throughout the entire training process.

The comparative learning progress of the three driving policies across four tracks is shown in Figure 9. The analysis of training episodes' termination reasons is shown in Figure 10. It is noteworthy that the termination categories are different from the time trial experiments: (1) the 'skid' termination category has been eliminated since the incorporation of AM in all evaluated policies effectively prevents loss-of-control scenarios caused by exceeding tire friction limits; (2) a new 'collision' category has been specifically introduced to episodes terminated due to contact between the ego car and opponent cars, representing a critical performance metric in competitive racing scenarios. The completion rates of TD3-AM, TD3-AMSM, and TD3-AMSMGP are 58.6%, 81.5%, and 92.5%, respectively, with TD3-AMSMGP achieving the highest completion rate.

The learning curves and termination reason statistics for the three driving policies across four tracks demonstrate consistent patterns in policy convergence and performance characteristics. All three policies exhibit stable learning progress, reaching convergence after around 6000 episodes (approximately 13 million training iterations), requiring roughly 6 h of training time. The additional computational cost introduced by state mapping and safe policy guidance modules remains minimal. Comparative analysis reveals that both the TD3-AMSM and TD3-AMSMGP methods, incorporating the SM technique, achieve significantly faster learning rates and higher cumulative rewards than the baseline TD3-AM approach on all four tracks. This performance advantage can be attributed to the substantially reduced terminations caused by collision in SM-enhanced methods, as evidenced in Figure 10. The SM mechanism enables safer overtaking maneuvers within predefined safety margins, effectively minimizing collisions with opponent cars and consequently yielding higher rewards through improved safety performance.

Further examination of termination statistics between TD3-AMSM and TD3-AMSMGP reveals the additional benefits of GP. The TD3-AMSMGP policy demonstrates markedly fewer off-track terminations, particularly during early training phases. This reflects GP's crucial role in guiding exploratory actions during initial training, thereby enhancing pol-

icy safety at the learning onset. This safety advantage manifests in the learning curves (Figure 9), where TD3-AMSMGP maintains superior performance to TD3-AMSM during the first 2000 episodes, primarily due to reduced penalty accumulation from off-track incidents. The combined evidence from both learning progression and termination analysis confirms that SM and GP effectively enhance overtaking safety and efficiency in the competitive racing scenarios.

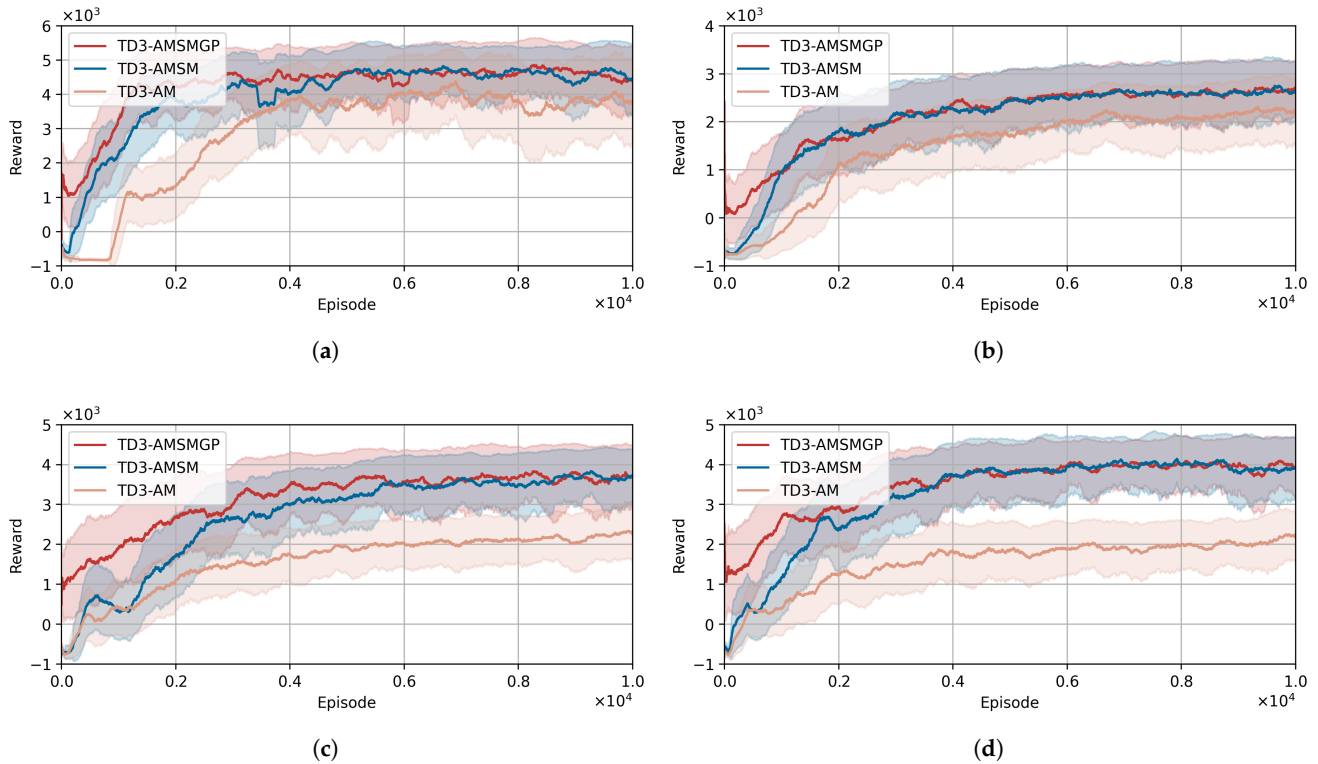


Figure 9. Learning curve comparison of three policies across four tracks in competitive racing scenario. (a) Learning curves in Track A. (b) Learning curves in Track B. (c) Learning curves in Track C. (d) Learning curves in Track D.

Consistent with the time trial experiments, we periodically save trained policies at 10,000-iteration intervals for subsequent evaluation. While the time trial assessment employs flying lap times to quantify basic driving skills, the competitive racing scenario necessitates a specialized evaluation framework. We therefore employ a Time Attack Overtake assessment paradigm. The Time Attack Overtake evaluation metric is designed to comprehensively assess competitive driving skills by measuring the maximum number of opponent cars overtaken within a fixed time window. The evaluation protocol initializes the ego car centered on the finish line with motion states $\omega = \phi = \delta = v = 0$, and the exploration noise is removed. The opponent cars are positioned at 80m intervals along the track centerline with randomized lateral offsets and a constant initial velocity of 40 km/h. All opponent vehicles are controlled by the same predetermined controller in the training phase. Each evaluation episode is constrained to 6000 time steps (60 s). We quantitatively evaluate all saved policies by recording both the overtaking count and average velocity. The best performance metrics (number of opponents overtaken and average velocity) for each policy–track combination are presented in Table 5. The results reveal that for the relatively simpler and wider tracks (A and B), all three driving policies demonstrate comparable performance in terms of both the number overtook and average velocity. However, in the more complex and narrower tracks (C and D), the two policies incorporating the state mapping (SM) mechanism exhibit significant performance advantages. Specifically, the SM-

enhanced policies achieve approximately double the number of successful overtakes within the time window compared to the baseline approach, while simultaneously maintaining about 50% higher average velocities.

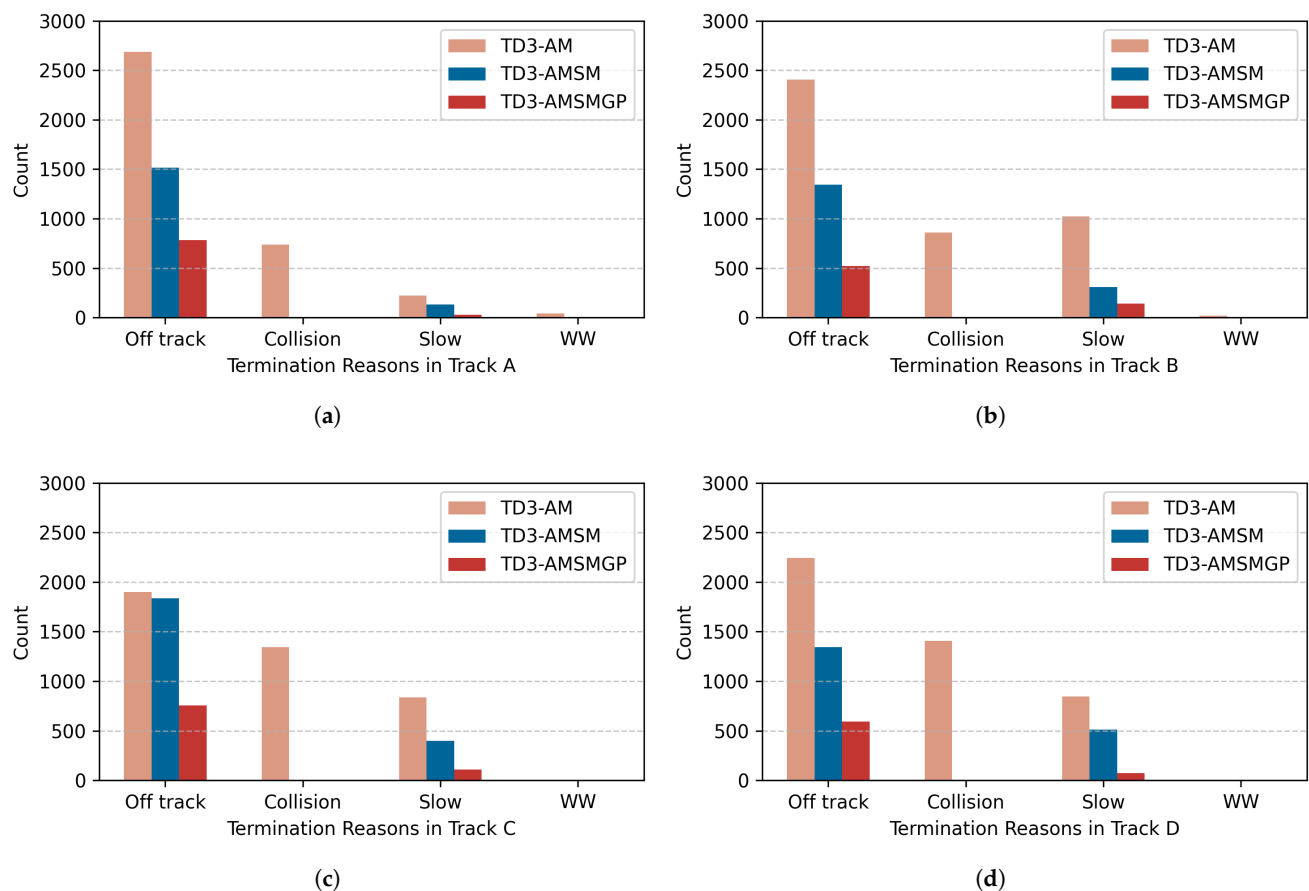


Figure 10. Count of termination reasons of 10,000 training episodes of three driving policies across four tracks in competitive racing scenario. (a) Count of termination reasons in Track A. (b) Count of termination reasons in Track B. (c) Count of termination reasons in Track C. (d) Count of termination reasons in Track D.

Table 5. Numbers of overtaking and average speed (inside bracket) of three driving policies across four tracks.

	Track A	Track B	Track C	Track D
TD3-AM	14 (92.9 km/h)	6 (53.3 km/h)	5 (46.8 km/h)	6 (54.4 km/h)
TD3-AMSM	15 (94.0 km/h)	7 (52.6 km/h)	10 (68.8 km/h)	12 (78.1 km/h)
TD3-AMSMGP	15 (95.4 km/h)	7 (56.9 km/h)	10 (70.9 km/h)	12 (79.6 km/h)

The racing trajectories of the highest-performing TD3-AMSMGP policy during Time Attack Overtake evaluations across the four tracks are illustrated in Figure 11. The ego car's trajectories are color-coded by instantaneous speed. The trajectories of opponent vehicles are depicted as gray lines, selectively displaying only those segments where potential interactions with the ego vehicle may occur. More precisely, for each opponent car, its trajectory visualization begins when it assumes the position of the immediate leading vehicle relative to the ego car and ceases once it has been successfully overtaken, thereby focusing the analysis on the most relevant spatial-temporal interactions during overtaking maneuvers. The trajectories demonstrate that the learned policy has successfully developed advanced overtaking capabilities while preserving the high-speed driving skills exhibited in time trials, particularly excelling in the complex, narrow configurations of Tracks C and

D, where it executes both straightaway and corner overtaking maneuvers with remarkable consistency. These results clearly validate the efficacy of the SM technique in facilitating competitive driving strategies.

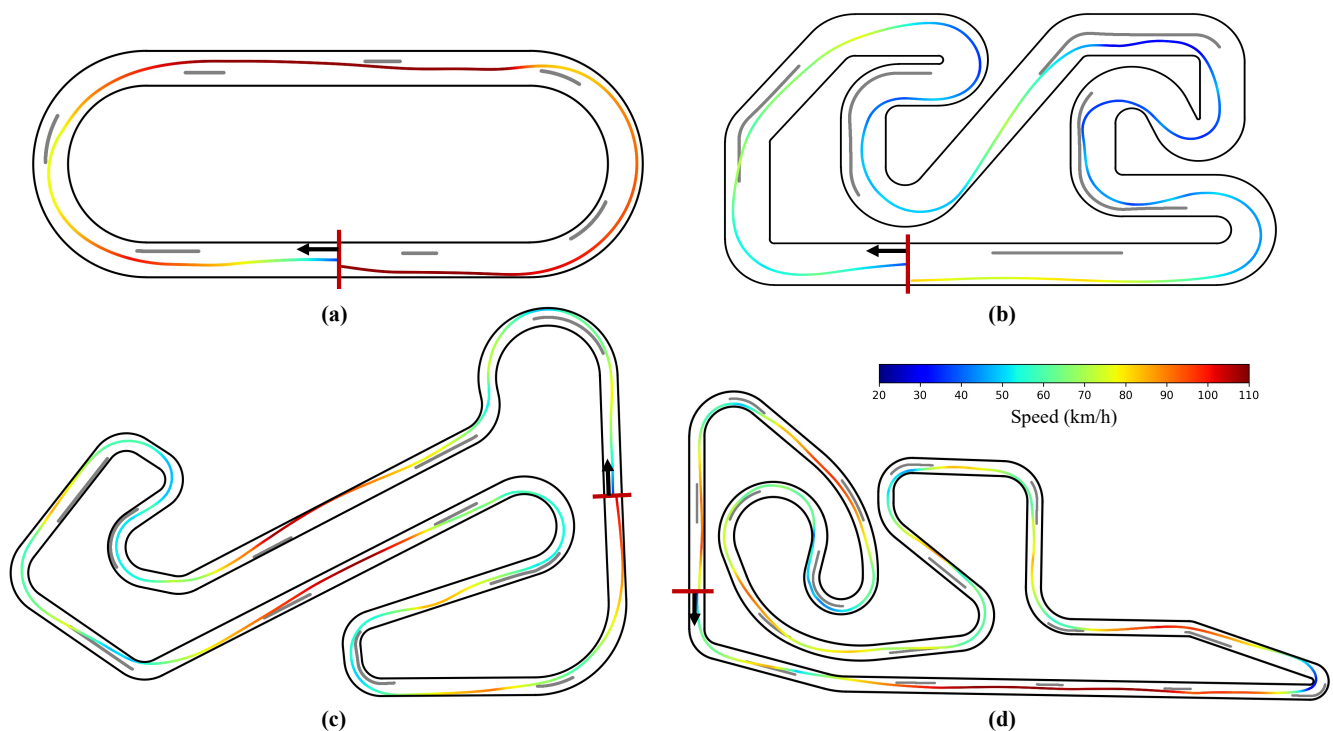


Figure 11. Time Attack Overtake trajectories of the TD3-AMSMGP driving policy across four tracks. The trajectories of the ego car and the opponent cars in Tracks A–D are depicted in subfigures (a–d), respectively. In each subfigure, the ego car’s trajectories are color-coded by instantaneous speed (warmer hues indicate higher velocities). Opponent cars’ trajectories are shown in gray lines that initiate when becoming the nearest leading vehicle and terminate upon being overtaken.

8.4. Discussion

The experimental results obtained from both time trial and competitive racing experiments demonstrate that the proposed RL-based driving policy can effectively learn high-performance driving strategies in environments with or without opponent vehicles. Through systematic training, the policy acquires professional-level cornering and overtaking skills while maintaining superior performance across diverse track configurations. The comparative studies of different policy configurations—incorporating AM, GP, and SM modules with the TD3 framework—respectively, validate (1) the AM module’s efficacy in handling tire–road friction constraints, (2) the GP module’s capability to enhance training safety during initial learning phases, and (3) the SM module’s effectiveness in improving competitive overtaking performance. Together, these components form a comprehensive autonomous racing solution that addresses the multifaceted challenges of high-speed vehicle control.

Despite the strong performance demonstrated, several limitations remain that could impact the real-world applicability and robustness of the proposed framework:

- The inherent trial-and-error nature of RL algorithm introduces safety concerns. Although a safe guidance policy is used to constrain exploration during training, exploratory noise remains necessary for learning. This can occasionally lead to off-track incidents, and the simulation environment cannot fully replicate real-world vehicle dynamics and environmental variability. Consequently, the safety guarantees achieved in simulation may not directly carry over to real-world deployments.

- The behavior of opponent vehicles is relatively simple. Currently, opponents follow predetermined trajectories and do not respond to the ego vehicle's actions. In real racing scenarios, opponents often exhibit adaptive behaviors, such as defending lines and blocking overtakes, which are crucial to competitive driving. This limitation reduces the policy's ability to learn complex interaction skills.
- The state mapping mechanism currently uses static parameters to define the overtaking feasibility areas. These parameters do not adapt based on the motion of surrounding vehicles. As a result, the policy may generate overly conservative or aggressive decisions during overtaking.

9. Conclusions and Future Work

This paper presented an integrated RL framework for autonomous race driving that combines state mapping, action mapping, and safe exploration guidance to achieve high-performance racing while ensuring safety. Experimental results across four test tracks demonstrate substantial improvements over baseline methods in both time trial and competitive racing scenarios. In time trial scenarios, the RL-based driving policy with action mapping and safe exploration guidance (TD3-AMGP) achieved a 12–26% reduction in lap time compared to the standard TD3 policy, while the completion rate improved significantly from 33.1% (TD3) to 78.7% (TD3-AMGP), indicating enhanced training safety. In competitive racing scenarios, the proposed TD3-AMSGP policy doubled the number of successful overtakes on complex tracks (Tracks C and D) compared to TD3-AM, while sustaining a 46–51% higher average speed. Across both scenarios, the inclusion of state mapping and the safe guidance policy effectively reduced off-track and collision-induced terminations by a large margin. These results confirm the proposed framework's effectiveness in achieving professional-level autonomous racing performance with a strong emphasis on safety and reliability.

Future work will focus on addressing the identified limitations through the following directions: (1) enhancing safety in real-world deployment by integrating RL policies with conventional control methods, such as model predictive control and control barrier functions, complemented by real-time risk assessment techniques; (2) improving competitive driving skills by incorporating interactive opponent behaviors via adversarial multi-agent RL or imitation learning, enabling the ego vehicle to acquire adaptive and tactical driving strategies; (3) developing dynamic state mapping mechanisms that adjust the parameters for generating overtaking feasible area in real time based on the opponent motion information; and (4) exploring sim-to-real transfer techniques such as domain randomization and adaptive simulation calibration to improve the robustness and deployability of learned policies in real-world racing environments.

Author Contributions: Conceptualization, Y.W. and X.Y.; methodology, Y.W. and X.Y.; software, Y.W. and J.L.; writing—original draft preparation, Y.W., J.L., and J.Y.; writing—review and editing, J.L. and J.Y.; visualization, Y.W.; funding acquisition, Y.W. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Natural Science Foundation of China under grant 62136008, in part by the Postdoctoral Fellowship Program of CPSF under grants 2024M750422 and GZB20240144, and in part by the Jiangsu Funding Program for Excellent Postdoctoral Talent under grant 2024ZB047.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Zhu, Z.; Zhao, H. A survey of deep RL and IL for autonomous driving policy learning. *IEEE Trans. Intell. Transp. Syst.* **2021**, *23*, 14043–14065. [\[CrossRef\]](#)
2. Betz, J.; Zheng, H.; Liniger, A.; Rosolia, U.; Karle, P.; Behl, M.; Krovi, V.; Mangharam, R. Autonomous vehicles on the edge: A survey on autonomous vehicle racing. *IEEE Open J. Intell. Transp. Syst.* **2022**, *3*, 458–488. [\[CrossRef\]](#)
3. He, S.; Zeng, J.; Sreenath, K. Autonomous racing with multiple vehicles using a parallelized optimization with safety guarantee using control barrier functions. In Proceedings of the 2022 International Conference on Robotics and Automation (ICRA), Philadelphia, PA, USA, 23–27 May 2022; pp. 3444–3451.
4. Mohseni, F.; Frisk, E.; Nielsen, L. Distributed cooperative MPC for autonomous driving in different traffic scenarios. *IEEE Trans. Intell. Veh.* **2020**, *6*, 299–309. [\[CrossRef\]](#)
5. Batkovic, I.; Gupta, A.; Zanon, M.; Falcone, P. Experimental validation of safe mpc for autonomous driving in uncertain environments. *IEEE Trans. Control Syst. Technol.* **2023**, *31*, 2027–2042. [\[CrossRef\]](#)
6. Jang, M.; Kim, S.; Yoo, B.; Oh, K. A Robust Path Tracking Controller for Autonomous Mobility with Control Delay Compensation Using Backstepping Control. *Actuators* **2024**, *13*, 508. [\[CrossRef\]](#)
7. Qiao, Y.; Chen, X.; Yin, D. Coordinated Control for the Trajectory Tracking of Four-Wheel Independent Drive–Four-Wheel Independent Steering Electric Vehicles Based on the Extension Dynamic Stability Domain. *Actuators* **2024**, *13*, 77. [\[CrossRef\]](#)
8. Jeong, Y. Integrated Vehicle Controller for Path Tracking with Rollover Prevention of Autonomous Articulated Electric Vehicle Based on Model Predictive Control. *Actuators* **2023**, *12*, 41. [\[CrossRef\]](#)
9. Kuutti, S.; Bowden, R.; Jin, Y.; Barber, P.; Fallah, S. A survey of deep learning applications to autonomous vehicle control. *IEEE Trans. Intell. Transp. Syst.* **2020**, *22*, 712–733. [\[CrossRef\]](#)
10. Xu, C.; Zhao, W.; Liu, J.; Wang, C.; Lv, C. An integrated decision-making framework for highway autonomous driving using combined learning and rule-based algorithm. *IEEE Trans. Veh. Technol.* **2022**, *71*, 3621–3632. [\[CrossRef\]](#)
11. Viadero-Monasterio, F.; Nguyen, A.T.; Lauber, J.; Boada, M.J.L.; Boada, B.L. Event-triggered robust path tracking control considering roll stability under network-induced delays for autonomous vehicles. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 14743–14756. [\[CrossRef\]](#)
12. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*; MIT Press: Cambridge, MA, USA, 2018.
13. Jaritz, M.; De Charette, R.; Toromanoff, M.; Perot, E.; Nashashibi, F. End-to-end race driving with deep reinforcement learning. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–26 May 2018; pp. 2070–2075.
14. Hell, M.; Hajgató, G.; Bogár-Németh, Á.; Bári, G. A lidar-based approach to autonomous racing with model-free reinforcement learning. In Proceedings of the 2024 IEEE Intelligent Vehicles Symposium (IV), Jeju Island, Republic of Korea, 2–5 June 2024; pp. 258–263.
15. Zhang, R.; Hou, J.; Chen, G.; Li, Z.; Chen, J.; Knoll, A. Residual policy learning facilitates efficient model-free autonomous racing. *IEEE Robot. Autom. Lett.* **2022**, *7*, 11625–11632. [\[CrossRef\]](#)
16. Fuchs, F.; Song, Y.; Kaufmann, E.; Scaramuzza, D.; Dürr, P. Super-human performance in gran turismo sport using deep reinforcement learning. *IEEE Robot. Autom. Lett.* **2021**, *6*, 4257–4264. [\[CrossRef\]](#)
17. Evans, B.D.; Jordaan, H.W.; Engelbrecht, H.A. Comparing deep reinforcement learning architectures for autonomous racing. *Mach. Learn. Appl.* **2023**, *14*, 100496. [\[CrossRef\]](#)
18. Wang, Y.; Yuan, X.; Sun, C. Learning autonomous race driving with action mapping reinforcement learning. *ISA Trans.* **2024**, *150*, 1–14. [\[CrossRef\]](#)
19. Gottschalk, S.; Gerdt, M.; Piccinini, M. Reinforcement Learning and Optimal Control: A Hybrid Collision Avoidance Approach. In Proceedings of the VEHITS, Angers, France, 2–4 May 2024; pp. 76–87.
20. Niu, J.; Hu, Y.; Li, W.; Huang, G.; Han, Y.; Li, X. Closing the dynamics gap via adversarial and reinforcement learning for high-speed racing. In Proceedings of the 2022 International Joint Conference on Neural Networks (IJCNN), Padua, Italy, 18–23 July 2022; pp. 1–10.
21. Liu, R.; Zhuang, W.; Tong, F.; Yin, G. Learn to race: Sequential actor-critic reinforcement learning for autonomous racing. In Proceedings of the 2023 IEEE International Automated Vehicle Validation Conference (IAVVC), Austin, TX, USA, 16–18 October 2023; pp. 1–6.
22. Chen, B.; Francis, J.; Oh, J.; Nyberg, E.; Herbert, S.L. Safe autonomous racing via approximate reachability on ego-vision. *arXiv* **2021**, arXiv:2110.07699.
23. Stachowicz, K.; Levine, S. Racer: Epistemic risk-sensitive rl enables fast driving with fewer crashes. *arXiv* **2024**, arXiv:2405.04714.
24. Francis, J.; Chen, B.; Ganju, S.; Kathpal, S.; Poonganam, J.; Shivani, A.; Vyas, V.; Genc, S.; Zhukov, I.; Kumskey, M.; et al. Learn-to-race challenge 2022: Benchmarking safe learning and cross-domain generalisation in autonomous racing. *arXiv* **2022**, arXiv:2205.02953.

25. Balaji, B.; Mallya, S.; Genc, S.; Gupta, S.; Dirac, L.; Khare, V.; Roy, G.; Sun, T.; Tao, Y.; Townsend, B.; et al. Deep racer: Autonomous racing platform for experimentation with sim2real reinforcement learning. In Proceedings of the 2020 IEEE international conference on robotics and automation (ICRA), Paris, France, 31 May–31 August 2020; pp. 2746–2754.
26. Cai, P.; Wang, H.; Huang, H.; Liu, Y.; Liu, M. Vision-based autonomous car racing using deep imitative reinforcement learning. *IEEE Robot. Autom. Lett.* **2021**, *6*, 7262–7269. [[CrossRef](#)]
27. Song, Y.; Lin, H.; Kaufmann, E.; Dürr, P.; Scaramuzza, D. Autonomous overtaking in gran turismo sport using curriculum reinforcement learning. In Proceedings of the 2021 IEEE international conference on robotics and automation (ICRA), Xi'an, China, 30 May–5 June 2021; pp. 9403–9409.
28. Yuan, Y.; Tasik, R.; Adhatarao, S.S.; Yuan, Y.; Liu, Z.; Fu, X. RACE: Reinforced cooperative autonomous vehicle collision avoidance. *IEEE Trans. Veh. Technol.* **2020**, *69*, 9279–9291. [[CrossRef](#)]
29. Thakkar, R.S.; Samyal, A.S.; Fridovich-Keil, D.; Xu, Z.; Topcu, U. Hierarchical control for head-to-head autonomous racing. *Field Robot.* **2024**, *4*, 46–69. [[CrossRef](#)]
30. Wurman, P.R.; Barrett, S.; Kawamoto, K.; MacGlashan, J.; Subramanian, K.; Walsh, T.J.; Capobianco, R.; Devlic, A.; Eckert, F.; Fuchs, F.; et al. Outracing champion Gran Turismo drivers with deep reinforcement learning. *Nature* **2022**, *602*, 223–228. [[CrossRef](#)] [[PubMed](#)]
31. Shrestha, E.; Wan, H.; Reddy, C.; Zhuang, Y.; Vasudevan, R. Multimodal Model-Based Reinforcement Learning for Autonomous Racing. In Proceedings of the Deployable RL: From Research to Practice @ Reinforcement Learning Conference 2024, Amherst, MA, USA, 9–12 August 2024.
32. Liu, K.; Gong, J.; Kurt, A.; Chen, H.; Ozguner, U. Dynamic modeling and control of high-speed automated vehicles for lane change maneuver. *IEEE Trans. Intell. Veh.* **2018**, *3*, 329–339. [[CrossRef](#)]
33. Gao, F.; Hu, Q.; Ma, J.; Han, X. A simplified vehicle dynamics model for motion planner designed by nonlinear model predictive control. *Appl. Sci.* **2021**, *11*, 9887. [[CrossRef](#)]
34. Rajamani, R. *Vehicle Dynamics and Control*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2011.
35. Yuan, X.; Wang, Y.; Liu, J.; Sun, C. Action Mapping: A Reinforcement Learning Method for Constrained-Input Systems. *IEEE Trans. Neural Netw. Learn. Syst.* **2023**, *34*, 7145–7157. [[CrossRef](#)]
36. Fujimoto, S.; Hoof, H.; Meger, D. Addressing Function Approximation Error in Actor-Critic Methods. In Proceedings of the International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; pp. 1587–1596.
37. Thrun, S.; Montemerlo, M.; Dahlkamp, H.; Stavens, D.; Aron, A.; Diebel, J.; Fong, P.; Gale, J.; Halpenny, M.; Hoffmann, G.; et al. Stanley: The robot that won the DARPA Grand Challenge. *J. Field Robot.* **2006**, *23*, 661–692. [[CrossRef](#)]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.