This paper has been accepted for publication in the Proceedings of the 2025
IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2025)

On learning racing policies with reinforcement learning

Grzegorz Czechmanowski, Jan Węgrzynowski, Piotr Kicki, Krzysztof Walas

***Abstract*— Fully autonomous vehicles promise enhanced safety and efficiency. However, ensuring reliable operation in challenging corner cases requires control algorithms capable of performing at the vehicle limits. We address this requirement by considering the task of autonomous racing and propose solving it by learning a racing policy using Reinforcement Learning (RL). Our approach leverages domain randomization, actuator dynamics modeling, and policy architecture design to enable reliable and safe zero-shot deployment on a real platform. Evaluated on the F1TENTH race car, our RL policy not only surpasses a state-of-the-art Model Predictive Control (MPC), but, to the best of our knowledge, also represents the first instance of an RL policy outperforming expert human drivers in RC racing. This work identifies the key factors driving this performance improvement, providing critical insights for the design of robust RL-based control strategies for autonomous vehicles.**

I. Introduction

Ensuring the reliable performance of fully autonomous vehicles in critical corner cases is essential to achieve their promised safety and enable large-scale deployment. In dynamic scenarios, even small control inaccuracies can lead to crashes. As a result, control strategies must be both robust to uncertainties and capable of operating at the vehicle's limit to minimize the risk of accidents.

Autonomous racing is an example of a task that enables a rigorous comparison of control algorithms operating at the limits of their capabilities. The narrow margins of error in racing require vehicles to navigate with exceptional precision, making the racetrack an ideal testbed for evaluating and refining advanced control strategies. To date, classical optimization-based approaches remain among the most widely used methods in autonomous racing [1]. Although these methods offer high performance and interpretability, even minor model inaccuracies when operating at the limits of handling can result in crashes.

One promising approach to developing robust control algorithms is the use of Reinforcement Learning (RL). Unlike optimization-based controllers, RL can leverage domain randomization to improve robustness against modeling errors and effectively handle sparse and discontinuous objectives. Moreover, RL is not constrained by a finite prediction horizon, allowing it to optimize long-term performance beyond the capabilities of traditional optimization-based methods. Previous studies have demonstrated that RL can surpass professional human drivers in car racing simulations [2], [3], [4]

All authors are with IDEAS Research Institute, Warsaw, Poland, IDEAS NCBR, Warsaw, Poland and with Institute of Robotics and Machine Intelligence, Poznan University of Technology, Poznan, Poland. name.surname@put.poznan.pl This research was partially funded by PUT internal grant 0214/SBAD/0252. Work of Piotr Kicki was supported by the Foundation for Polish Science (FNP).

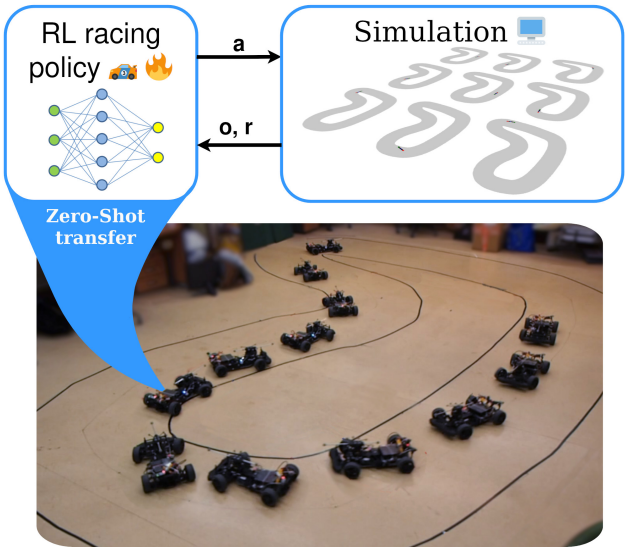

Fig. 1. Our RL policy, trained entirely in simulation, deployed zero-shot on an F1TENTH car in the real world.

and even outperform expert pilots in real-world drone racing [5]. However, despite its success in racing simulations, RL policies have struggled to transfer effectively to real-world scenarios. Consequently, optimization-based methods have consistently outperformed RL policies in real-world car racing [6], [7].

In this work, we propose a novel RL-based approach to autonomous racing that not only outperforms state-of-the-art MPC methods but also, to the best of our knowledge, represents the first instance of a RL policy that surpasses an expert human driver in RC racing. Notably, our benchmark human driver achieved fifth place in the Polish Indoor RC Championship, validating the competitive standard against which our autonomous system was evaluated. We systematically explore critical design choices, including domain randomization, actuator dynamics modeling, and policy architecture, which are essential to improve performance, safety, and transferability. The RL policies are trained exclusively in simulation and subsequently validated on the F1TENTH platform [8] under real-world conditions without additional training. Supplementary video materials from the experimental evaluation can be found at https://grzegorzczput.github.io/rl-racing/.

Our contributions can be summarized as follows:

1) We systematically investigated the impact of domain randomization, actuator dynamics modeling, observation, and action space design on the performance, safety, and transferability of reinforcement learning policies for autonomous racing.

2) We evaluated our approach against the state-of-the-art MPC and an expert human RC driver in the real world F1TENTH racing, demonstrating a fastest lap time improvement of $0.078\,\mathrm{s}$ over MPC and $0.286\,\mathrm{s}$ over the expert human driver on a challenging $17\,\mathrm{m}$ racetrack (see Fig. 1).

II. Related work

A. Model Predictive Control

One of the most popular approaches to designing a controller for autonomous car racing is to leverage MPC algorithms [9], [10], [11]. These methods have proven to be effective in real-world deployments. However, a major drawback of optimization-based approaches is their limited control frequency, which is constrained by model complexity and horizon length. This limitation forces users to adopt relatively simple models and significantly shorter planning horizons than a full lap of a racing track.

Additionally, to ensure fast optimization convergence, the objective function typically includes auxiliary regularization terms [11], [12] or requires simplifications, such as restricting costs to quadratic penalties relative to a predefined trajectory [13]. As a result, the objective function often acts as a proxy, rather than directly optimizing for the controller's true objective.

B. Reinforcement Learning in Autonomous Racing

On the other hand, RL offers a promising solution to overcome these limitations. Reinforcement learning has been successfully applied to complex robotic systems such as quadrupeds [14], [15] and drones [16], [5], demonstrating its capability to handle high-dimensional models and long-horizon planning. Unlike MPC, where the computational burden occurs during inference, RL shifts the computational effort to the training phase, enabling efficient real-time execution once training is complete. Moreover, RL is well-suited for optimizing tasks with sparse and delayed rewards, such as minimizing lap times in drone racing [17].

The application of RL in autonomous car racing has been studied primarily in simulation [18], [19] and has even been shown to outperform professional human simulator drivers [4]. Nevertheless, implementing end-to-end RL policies in real-world racing remains a non-trivial task. As demonstrated in [6], researchers could not directly train a high-performance policy without the foundation of a geometric controller. Even with this foundation, the resulting policy still failed to achieve performance comparable to model-based approaches such as MPC. Similarly, despite employing real-world policy fine-tuning, the authors [7] were unable to exceed the performance benchmarks set by optimization-based controllers.

III. Materials and methods

A. Problem definition

In this paper, we consider the problem of real-world autonomous racing with an F1TENTH car. In general, this is an extremely complex problem that typically requires the integration of multiple subsystems responsible for aggregating the data from sensors, performing localization, opponent detection, planning, and control. However, in this paper, we focus solely on the control problem in a time-trial scenario, assuming perfect knowledge of the vehicle state. The goal of this task is to find a controller that, based on the knowledge of the vehicle's state and the racetrack, is able to generate control signals, allowing the car to maximize the progress along the track centerline while remaining within track boundaries. This objective may be considered as an accurate dense approximation of the sparse minimum lap time goal.

B. Proposed solution

To address the problem of autonomous racing in the time-trial scenario, we begin by developing an accurate simulation of the F1TENTH car. Our goal is to precisely model all relevant aspects of vehicle dynamics, including actuator dynamics and tire characteristics that are lacking in other simulators [20]. We identify system parameters using a long-horizon prediction loss, which helps stabilize the learned models.

With an accurate simulation in place, we leverage RL to train an autonomous racing policy in simulation. Since simulation models are never a perfect representation of reality, deploying the learned policy directly presents challenges due to model mismatches. To enhance sim-to-real transfer, we employ domain randomization, ensuring the policy is robust to simulator inaccuracies and unmodeled behaviors it may encounter in the real world.

C. Hardware setup

In this work, we utilized the F1TENTH platform, built on the Xray GTXE'22—a 1/8th-scale RC car equipped with four-wheel drive and powered by a single motor. Additionally, we equipped the platform with a Dynamixel servo mechanism for the steering column, allowing us to directly measure the steering angle and tune the PID of the position controller. The vehicle operates in two distinct control modes:

- **Wheel Speed Control Mode**: The control input is defined as $a = [\delta_{\text{ref}}, \omega_{\text{ref}}]^T$, where δ_{ref} is the reference steering angle sent directly to the servo for steering

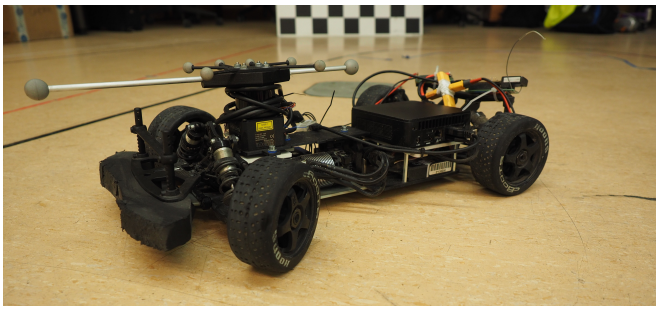

Fig. 2. F1TENTH car used for dataset colection and performing real-world experiments.

actuation, and ω_{ref} is the reference wheel speed, which is sent to the Electronic Speed Controllers (ESC). The ESC utilizes an internal PID controller to regulate the vehicle's wheel speed accordingly.

- **Motor Current Control Mode**: The control input is given by $a = [\delta_{\text{ref}}, I]^T$, where the steering control operates identically to the wheel speed control mode. However, instead of controlling wheel speed through the ESC, the motor is directly actuated by setting the current I.

The two control modes are designed to meet the distinct requirements of human drivers and autonomous control algorithms. Human drivers naturally regulate the vehicle by modulating the throttle signal, which in electric vehicles is typically implemented as motor current control. In contrast, autonomous control systems favor wheel speed regulation using high-frequency, low-level PID controllers inside ESC.

To enable both system identification and policy deployment, we employ an OptiTrack motion capture system. OptiTrack provides precise real-time measurements of the vehicle's position and orientation, which are then processed to estimate velocity in the vehicle frame. The velocity is computed by differentiating the position and orientation data using a Savitzky-Golay filter [21].

All training and real-world experiments were conducted on a single L-shaped track (see Fig. 1), measuring 17 meters in length and 1 meter in width. Despite being constrained by the OptiTrack tracking area, this layout incorporates both a high-speed section and a tight technical sector, providing a balanced environment for evaluating controllers' performance. This challenging setup ensures a comprehensive assessment of reinforcement learning policies and traditional control algorithms.

D. Simulation environment

The simulation environment is a key aspect of the reinforcement learning setup, especially for high-performance racing policies. In this paper, we focus solely on the accurate and fast simulation of racing car dynamics, completely neglecting the visual aspect of the simulation, as it will only be used for policy training. Therefore, we want the dynamics model to be (i) fast to evaluate, (ii) difficult or near impossible to exploit, (iii) accurate enough to enable racing at the limit of grip, and finally (iv) close to the hardware setup we use. Such qualities are beyond the scope of general-purpose simulators but can be obtained by using an analytical physics-derived model optimized to be consistent and accurate in long-horizon simulation.

In this paper, we used a single track vehicle dynamics model with the MF6.1 tire model [22], which extends the traditional Pacejka formulation [23] commonly used in autonomous racing [24] by jointly considering lateral and longitudinal tire slip to calculate tire forces. Moreover, to account for the dynamics of the physical actuators, we model them as a first-order system, identifying the relationship between the reference input and the measured position and velocity by

$$\begin{aligned} \dot{\delta}(t) &= \frac{1}{T_\delta}(\delta_{\text{ref}}(t) - \delta(t)), \\ \dot{\omega}(t) &= \frac{1}{T_\omega}(\omega_{\text{ref}}(t) - \omega(t)), \end{aligned} \tag{1}$$

where T_δ and T_ω are the time constants T_δ for the steering actuator and the approximation of the PID regulators inside ESC, respectively.

We identified the parameters of the introduced model using the dataset collected with the physical setup introduced in Section III-C. This dataset was collected by an expert human driver operating across various tracks, intentionally varying slip angles and slip ratios to cover the full operational envelope of the vehicle's state space. During data collection, we captured relevant data about the car's internal state, such as motor rotational speed and steering angle, as well as information about its position and orientation using the OptiTrack motion capture system. All of the data were collected synchronously at 100Hz. In total, we recorded 36 minutes of driving.

Finally, similarly to [25], we formulated the Mean Squared Error (MSE) loss between the state trajectory roll-out obtained using the dynamics model and ground-truth data on sequences of 85 samples. To minimize this loss, we used stochastic gradient descent optimization, in particular AdamW [26], and backpropagated the loss gradient with respect to the dynamics model's parameters through time.

E. Reinforcement Learning Training Procedure

Leveraging the accurate simulation environment detailed in Section III-D, we train our RL policy using the Proximal Policy Optimization (PPO) algorithm [27], [28], in a parallelized setup with 400 simulated instances running concurrently. Each rollout consists of 1024 steps, with a time step of 0.05s. The training was performed using a batch size of 1024. If the agent exceeds track limits during a rollout, it is reset to a random position on the track's centerline to enhance the exploration. During training, the parameters of the dynamics model are randomized at the beginning of each episode by applying multiplicative perturbations with Gaussian noise. Specifically, each parameter θ is scaled by a factor drawn from $\mathcal{N}(1, \sigma_{dr})$, where σ_{dr} is specified in the experimental configuration.

The sum of discounted rewards is calculated with a discount factor of $\gamma = 0.99$. The learning rate decays from 1×10^{-3} to 1×10^{-4} over the course of training. The total training process spans 120 million environment steps, allowing the agent to experience a wide range of driving scenarios and refine its policy accordingly.

Both the actor and critic networks are implemented as multilayer perceptrons. The actor network consists of two hidden layers, each with 256 neurons, while the critic network comprises two hidden layers with 512 neurons each. Both networks employ the Leaky Rectified Linear Unit activation function with a negative slope of 0.2.

The observation configuration for both networks is defined as follows: 1) Linear velocities in vehicle frame: longitudinal

v_x and lateral v_y. 2) Yaw rate: $\mathfrak{r}$. 3) The angle between the car's heading and the track heading, expressed in Frenet coordinates u. 4) Lateral distance from the centerline, in Frenet coordinates: n. 5) Measured steering angle δ and the previous control input δ_{ref}. 6) Measured wheel speed ω, commanded wheel speed ω_{ref}, and the last control input $\dot{\omega}_{\text{ref}}$. 7) Track information, represented by N points, with curvature $\boldsymbol{c} \in \mathbb{R}^N$ and width $\boldsymbol{w} \in \mathbb{R}^N$. These N points are sampled uniformly in front of the vehicle at 30-centimeter intervals. Thus, the observation vector is given by:

$$\mathbf{obs} = [v_x, v_y, u, n, \mathfrak{r}, \delta, \delta_{\text{ref}}, \dot{\omega}_{\text{ref}}, \omega_{\text{ref}}, \omega, \boldsymbol{c}, \boldsymbol{w}]. \tag{2}$$

Observations are normalized by dividing each element by its corresponding maximum value. The maximum values for state observations are derived from the system identification dataset, while those for track observations are computed based on the track geometry.

The final output of the actor network is a two-dimensional vector $[\delta_{\text{ref}}, \dot{\omega}_{\text{ref}}]^T$, with each element constrained to the interval $[-1, 1]$. These outputs are subsequently scaled by a factor of 0.5 for δ_{ref} and 5 for $\dot{\omega}_{\text{ref}}$ to yield the action vector. The steering scaling factor was chosen because 0.5 rad is the maximum limit of the steering actuator. The acceleration limit for the wheel was set to 5 m/s^2, as this was the highest longitudinal acceleration measured from the identification dataset. The reference steering angle δ_{ref} is directly applied to the servo for steering actuation, while the reference wheel speed derivative $\dot{\omega}_{\text{ref}}$ is integrated over time, analogously to [29], before being passed to the wheel speed controller. This formulation of speed control mitigates abrupt velocity changes, thereby enhancing policy transferability to the real world.

To effectively train racing policies, we designed the reward function to quantify the vehicle's progress along the track and penalize boundary violations, and formulated it as follows:

$$r_t = \begin{cases} -1, & \text{if the track boundary is exceeded,} \\ \mathfrak{s}_t - \mathfrak{s}_{t-1}, & \text{otherwise,} \end{cases} \tag{3}$$

where $\mathfrak{s}_t$ denotes the progress along the centerline expressed in Frenet coordinates [30]. Although the primary goal is to achieve faster laps, designing the reward function directly around this metric would hinder learning due to its sparse nature. Therefore, progress along the centerline is used as an effective proxy that provides denser feedback. This is why such a reward formulation, or similar variants, is commonly employed in other RL racing problems [3], [4], [5].

F. Bridging sim-to-real gap

An essential aspect of transferring policies trained in simulation to real-world scenarios is to utilize accurate dynamics models. Nevertheless, even with precise system modeling and identification, discrepancies between simulation and reality are inevitable and can compromise both safety and performance when deploying a policy in real-world conditions. To address this challenge, domain randomization [31] is often employed during training by perturbing the model parameters. This process enhances the robustness of the learned policy across a range of dynamics models, thereby reducing overfitting to the simulated environment. By training on the distribution of models, the policy becomes less sensitive to specific parameter values and more resilient to unforeseen variations in real-world dynamics, ultimately facilitating a safer and more reliable transfer from simulation to reality. An important consideration when using domain randomization to enhance robustness is the trade-off between performance and resilience. Although increasing randomization can lead to a more robust policy, it ultimately limits performance because the policy prioritizes conservatism over exploitation, as we show in Section IV-A.

IV. Experiments

To determine the key factors that affect learning RL policies for autonomous racing, we conducted a series of experiments. A major focus of our study was to address the sim-to-real gap, for which we evaluated various domain randomization strategies (see Section IV-A) and the impact of actuator modeling (see Section IV-D) to identify the most effective approach to improve real-world transferability.

Additionally, we investigated the impact of action and observation space choices on policy performance (see Sections IV-B and IV-C), aiming to understand how different representations influence policies' behavior when tested in real-world conditions.

Finally, we compare the introduced RL racing agent with the MPC baseline and refer their results to those obtained by an expert human driver (see Section IV-F).

A. Domain randomization

To evaluate the impact of domain randomization on both policy performance and safety, we trained networks using varying levels of parameter perturbation, specifically with $\sigma \in \{0.0, 0.02, 0.05, 0.1\}$. Two randomization strategies were considered: (i) randomizing only the tire-track friction, acknowledging that friction can vary due to small changes in temperature and amount of dust on the track, and (ii) randomizing all single-track parameters (vehicle mass, inertia, weight distribution, etc.) to increase the policy's robustness against model mismatch.

Policy performance was assessed by comparing both lap times and rack boundaries violations, which was quantified by the time-integral of the off-track distance, averaged per lap, measured in $[m \cdot s]$. The off-track distance is defined as

$$e_{\text{off}}(t) = \begin{cases} |n(t)| - W_{\mathfrak{s}(t)}, & \text{if } |n(t)| > W_{\mathfrak{s}(t)}, \\ 0, & \text{otherwise,} \end{cases} \tag{4}$$

where $W_{\mathfrak{s}}$ denotes the track width at the corresponding point $\mathfrak{s}$. The integrated off-track error is given by

$$E_{\text{off}} = \frac{1}{N} \int_0^T e_{\text{off}}(t)\, dt, \tag{5}$$

where N represents the number of laps driven, and T represents the duration of the test episode. During testing, each domain randomization configuration was evaluated over 20 laps (i.e., $N = 20$).

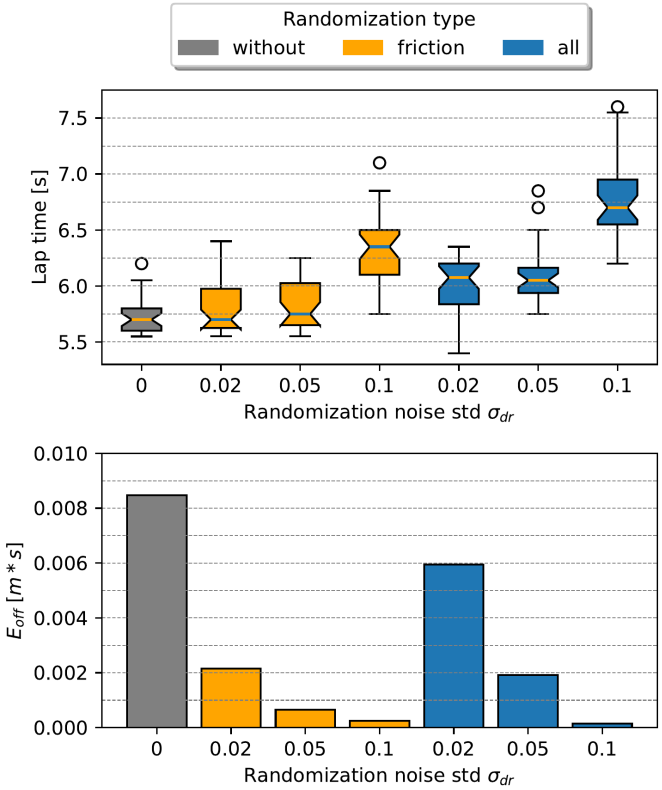

Fig. 3. Real-world performance of networks trained with randomized friction and single-track parameters. **Top:** Lap times achieved with different types and σ_{dr} of randomization. **Bottom:** E_{off} and percentage of laps with crashes for different randomization configurations.

The results of the experiment with randomized friction levels and single track parameters are presented in Figure 3. As friction randomization increases, lap times tend to increase, while the integrated off-track error decreases. A similar effect was observed when randomizing all single-track parameters; however, overall, single-track parameter randomization caused worse performance, resulting in longer lap times and higher E_{off} compared to policies trained with only friction randomization. This suggests that under more significant parameter variability, the policy adopts a more conservative driving strategy to remain valid across the entire distribution of dynamic models encountered during training. While domain randomization enhances robustness, excessive perturbations across all parameters can degrade overall performance. When the policy is trained over a wide distribution of models, it is forced to learn a control strategy that generalizes across the entire range of dynamics rather than optimizing for a specific one. Since the policy has no access to action history or recurrent memory to infer the underlying system dynamics at a given moment, it cannot adapt its behavior to a particular model instance. Instead, it must adopt a more cautious, suboptimal racing strategy that remains feasible across all possible parameter variations, ultimately leading to longer lap times.

These findings highlight the necessity of balancing domain randomization to ensure both safety and real-world transferability. Even with precise system identification, a moderate level of randomization is beneficial for mitigating simulation-to-reality discrepancies and improving policy robustness. However, while limited domain randomization improves generalization, high-performance policies still require accurate system identification, as randomization alone cannot fully compensate for an inaccurate dynamics model. For subsequent experiments, we selected the policy trained with friction randomization at $\sigma_{dr} = 0.02$, and we refere to it as the base policy. This policy was selected for its acceptable trade-off between speed and robustness.

B. Track representation

A key aspect of training an effective racing policy is the choice of track representation, which influences learning efficiency and generalization capabilities. Two commonly used approaches to represent track information in RL policies are:

1) **Geometric Representation:** The track is represented using curvature profiles ($\boldsymbol{c}$), track widths ($\boldsymbol{w}$), or range measurements to track boundaries. This provides the policy with information about the upcoming track layout, allowing it to generalize learned behaviors across similar sections. However, this representation has a limited prediction horizon, as it only provides information about the next meters of track, and the policy must simultaneously learn to localize itself on the track from these measurements.
2) **Track progress representation:** The track information is represented in terms of the Frenet coordinate s, which represents progress along the centerline. This approach ensures that the policy always has precise information about its location on the track. However, unlike the geometric representation, it lacks explicit information about the upcoming track shape, limiting the policy's ability to generalize across similar sections.

To investigate the impact of track representation on policy performance, we trained policies using both approaches: (i) the defaul geometric representation **obs** defined in (2) and an alternative (ii) track progress representation $\mathbf{obs}_{\mathfrak{s}}$ defined by

$$\mathbf{obs}_{\mathfrak{s}} = [n, u, v_x, v_y, \mathfrak{r}, \delta, \delta_{\text{ref}}, \dot{\omega}_{\text{ref}}, \omega_{\text{ref}}, \omega, \mathfrak{s}]. \tag{6}$$

Despite the differences in input structure, both policies successfully converged to the same reward levels, as shown in Fig. 4. This suggests that, with sufficient training, RL policies can extract the necessary spatial and temporal information from either representation.

Both policies were deployed in real-world experiments, where they achieved comparable performance. However, as presented in Table I, the policy utilizing geometric track representation (**obs**) achieved slightly faster lap times.

C. Action space

Another critical policy design choice is the action space selection, which can directly impact both performance and sim-to-real transfer. We evaluated the effect of control space selection on policy performance and sim-to-real transfer.

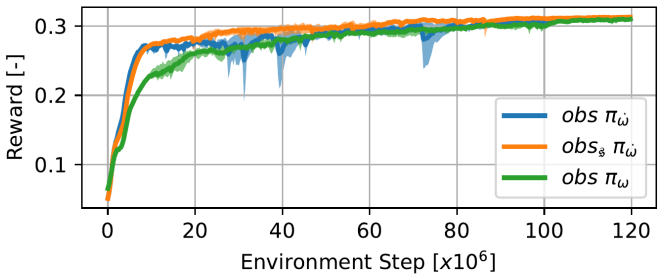

Fig. 4. Learning curves of the base policy using $\pi_{\dot{\omega}}$ with obs, the policy using $\mathfrak{s}$ for track representation $obs_{\mathfrak{s}}$, and the wheel speed control policy π_{ω}.

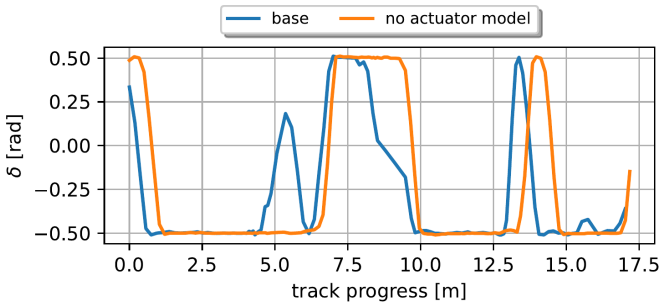

Fig. 5. Sterring control of networks trained with and without modeled actuators dynamics.

Using an otherwise identical training configuration, we compared two control architectures: one in which the policy π_{ω} directly controls vehicle wheel speed ω_{ref} and another in which it modulates the acceleration of the clamped wheel speed $\dot{\omega}_{ref}$. In the simulation, the velocity control policy converged to performance levels comparable to the acceleration control policy, as demonstrated in Figure 4. However, in real-world tests, the wheel speed control-based policy proved overly aggressive, frequently exceeding track boundaries by more than 20 cm and failing to complete 10 laps without crashing. These results underscore the critical importance of choosing a control space that limits abrupt control actions, thus enhancing the robustness of sim-to-real transfer.

D. Actuator modeling

Actuator dynamics is often neglected in vehicle dynamics modeling. In our work, both the powertrain and the steering actuators are modeled as first-order systems, as described in Equation (1). To assess the impact of actuator modeling on policy performance, we trained a policy without modeled actuator dynamics, deployed it in the real world, and presented its performance in Table I.

The policy trained without modeled actuators RL w/o *am* is significantly slower than the baseline RL policy (see Table I). As illustrated in Figure 5, the policy with modeled actuators initiates steering adjustments earlier, effectively compensating for actuator dynamics, while the baseline policy exhibits delayed responses. This finding underscores the importance of incorporating accurate actuator models to improve policy performance and enable more reliable real-world deployment.

TABLE I

COMPARISON OF FASTEST LAP TIMES, AVERAGE LAP TIMES, AND E_{OFF}. OVER A 20-LAP EXPERIMENT.

Name	Fastest Lap* (s)	Avg Lap Time (s)	E_{off} (m∗s)
RL **obs**	**5.561**	**5.768±0.10**	0.00216
RL w/o *am***	6.430	6.785±0.22	0.00118
RL **obs**$_{\mathfrak{s}}$	5.598	5.812±0.12	0.00178
MPC	5.639	5.804±0.17	**0.00083**
Human	5.847	5.939±0.19	0.01970

* fastest lap time recorded without exceeding track boundaries.
** RL policy trained without actuator modeling.

E. MPC baseline

To evaluate our RL based controller against other state-of-the-art approaches, we implemented MPC using acados [32] with a Frenet frame formulation following [33], [12], [34]. The controller's objective was to maximize progress along the center line with a soft constraint on track boundaries, similar to [35]. We applied regularization $q_{\beta}(\beta_{kin} - \beta_{dyn})^2$ on the difference between kinematic and dynamic single track model slip angles as in [12], [34]. To limit the controller's aggressiveness, we incorporated additional action smoothness costs by adding $q_{\delta}(\delta_{\text{ref}} - \delta)^2 + q_{\omega}(\omega_{\text{ref}} - \omega)^2$ to the primary objective. The controller used 80 shooting nodes with a 2.67s horizon, and we tuned the weights $q_{\beta}, q_{\delta}, q_{\omega}$ for optimal performance. To compensate for computational delays, the MPC was initialized from the predicted state.

The MPC controller was implemented with an objective that was as similar to the RL policy as possible, with minimal addition of auxiliary losses to ensure real-time optimization convergence. Furthermore, the dynamics model used in the MPC was identical to the one in the simulation environment, resulting in our best-performing MPC implementation.

F. Policy performance

To evaluate our racing policy, we compared its performance against a state-of-the-art MPC approach and an expert human driver, who placed fifth in the Polish Indoor RC Championships.

To ensure a fair comparison between the autonomous approaches and the human driver, we provided the expert with the track layout a week before the test. The driver was allowed to test the car in both the Motor Current Control Mode and Wheel Speed Control Mode. After evaluating both options, he selected Current Control Mode and was allowed to adjust the maximum current limit to match his driving style.

Following this setup, he was given two hours of practice to familiarize himself with both the car and the track. He controlled the car from an elevated position above the track surface to ensure optimal visibility.

After the practice session, the driver was instructed to complete 20 laps as quickly as possible while staying within track boundaries. The performance metrics of the human driver, our RL policy, and the MPC approach are summarized in Table I.

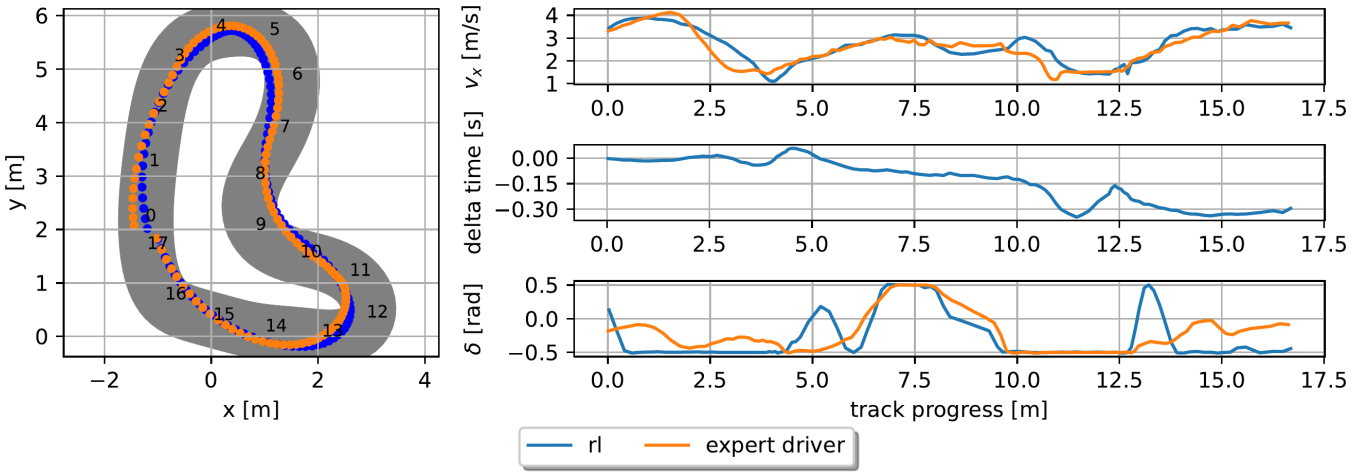

Fig. 6. Comparison of the fastest recorded lap without exceeding track boundaries by a human expert driver and the RL policy. **Left:** Trajectories driven on the track, with black numbers indicating progress along the center line $\mathfrak{s}$. **Right:** From top to bottom, the panels display the car's longitudinal velocity (v_x) during the lap, the delta time between the RL policy and the expert human driver, and the variation in steering command.

The RL policy outperformed the MPC baseline in both fastest-lap and average-lap times. We attribute RL's advantage, at least in part, to MPC's finite prediction horizon, which biases control toward locally optimal actions rather than lap-time–optimal trajectories. Moreover, the RL policy surpassed the expert human driver across all evaluated metrics.To the best of our knowledge, this is the first time that the RL policy has exceeded expert human performance in RC car racing. Notably, MPC achieved the lowest $\boldsymbol{E}_{\text{off}}$, while the human driver recorded the highest. This discrepancy may be attributed to the track's relatively small size compared to the vehicle and the low tire-road friction conditions, which differed significantly from those typically encountered in RC racing competitions. Unlike traditional RC race tracks, which are often carpeted to enhance grip, our test track's slippery surface may have posed an unexpected challenge for the human driver, limiting their ability to fully exploit the vehicle's handling capabilities.

The racing lines followed by the human driver and the RL policy — illustrated in the left image of Figure 6 — are similar. Although the RL policy initially takes a tighter arc from $\mathfrak{s} = 0\ m$ to $\mathfrak{s} = 5\ m$, it does not immediately gain time over the human driver, but instead positions the vehicle more favorably for the upcoming sections. Between $\mathfrak{s} = 5\ m$ and $\mathfrak{s} = 11\ m$, while both maintain the same longitudinal velocity (v_x), the RL policy gains approximately 0.14 seconds by adopting a more efficient racing line. In the final segment, from $\mathfrak{s} = 11\ m$ to $\mathfrak{s} = 17\ m$, the policy carries more speed into the corner, securing an additional 0.15-second advantage, resulting in a total lead of nearly 0.3 seconds over the expert human driver for the entire lap.

This comparison reinforces the notion that RL-based methods can surpass not only expert human drivers but also state-of-the-art optimization-based controllers. These results highlight the potential of reinforcement learning as a viable alternative for high-performance autonomous racing and, more broadly, for real-world applications that require control at the vehicle limits.

V. Conclusion

In this work, we investigated the applicability of reinforcement learning to real-world autonomous racing using a scaled car, evaluating its ability to operate at the limits of vehicle dynamics while maintaining safety and transferability to real-world deployment. Our approach systematically explored the impact of domain randomization, actuator modeling, and policy architecture on both performance and robustness. To the best of our knowledge, we demonstrated that our RL policy is the first to surpass state-of-the-art MPC and expert human driver in RC car racing.

Our findings highlight several key insights for RL-based racing policies. First, moderate domain randomization improves robustness and sim-to-real transferability; however, excessive perturbations across all parameters degrade performance by forcing the policy to generalize too broadly. Additionally, accurate actuator modeling is critical for achieving high performance, as it enables the policy to anticipate actuator dynamics and optimize control actions accordingly. Finally, selecting an appropriate control architecture plays a crucial role in stability, with action spaces that limit abrupt control changes, leading to safer and more effective real-world policies.

Despite these advancements, several challenges remain. Although our policy demonstrates superior performance in single-lap racing, future work should focus on adapting to dynamically changing track conditions and incorporating real-time model adaptation. Moreover, further exploration of hybrid learning-based and model-based approaches may yield even greater improvements in performance and safety.

Our results underscore the potential of RL for high-performance autonomous racing and provide valuable guidelines for future research in safe and efficient learning-based control strategies.

REFERENCES

[1] J. Betz *et al.*, "Autonomous vehicles on the edge: A survey on autonomous vehicle racing," *IEEE Open Journal of Intelligent Transportation Systems*, vol. 3, pp. 458–488, 2022.

[2] P. R. Wurman *et al.*, "Outracing champion Gran Turismo drivers with deep reinforcement learning," *Nature*, vol. 602, no. 7896, pp. 223–228, Feb. 2022, publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/s41586-021-04357-7

[3] Y. Song, H. Lin, E. Kaufmann, P. Duerr, and D. Scaramuzza, "Autonomous Overtaking in Gran Turismo Sport Using Curriculum Reinforcement Learning," May 2021, arXiv:2103.14666 [cs]. [Online]. Available: http://arxiv.org/abs/2103.14666

[4] F. Fuchs, Y. Song, E. Kaufmann, D. Scaramuzza, and P. Duerr, "Super-Human Performance in Gran Turismo Sport Using Deep Reinforcement Learning," *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 4257–4264, Jul. 2021, arXiv:2008.07971 [cs]. [Online]. Available: http://arxiv.org/abs/2008.07971

[5] E. Kaufmann *et al.*, "Champion-level drone racing using deep reinforcement learning," *Nature*, vol. 620, no. 7976, pp. 982–987, Aug. 2023. [Online]. Available: https://www.nature.com/articles/s41586-023-06419-4

[6] E. Ghignone *et al.*, "Rlpp: A residual method for zero-shot real-world autonomous racing on scaled platforms," *arXiv preprint arXiv:2501.17311*, 2025.

[7] E. Chisari, A. Liniger, A. Rupenyan, L. Van Gool, and J. Lygeros, "Learning from simulation, racing in reality," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 8046–8052.

[8] M. O'Kelly, H. Zheng, D. Karthik, and R. Mangharam, "F1tenth: An open-source evaluation environment for continuous control and reinforcement learning," in *Proceedings of the NeurIPS 2019 Competition and Demonstration Track*, ser. Proceedings of Machine Learning Research, H. J. Escalante and R. Hadsell, Eds., vol. 123. PMLR, 08–14 Dec 2020, pp. 77–89. [Online]. Available: https://proceedings.mlr.press/v123/o-kelly20a.html

[9] A. Liniger, A. Domahidi, and M. Morari, "Optimization-based autonomous racing of 1: 43 scale rc cars," *Optimal Control Applications and Methods*, vol. 36, no. 5, pp. 628–647, 2015.

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

[11] J. Kabzan, L. Hewing, A. Liniger, and M. N. Zeilinger, "Learning-Based Model Predictive Control for Autonomous Racing," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3363–3370, Oct. 2019.

[12] S. Srinivasan, S. N. Giles, and A. Liniger, "A Holistic Motion Planning and Control Solution to Challenge a Professional Racecar Driver," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7854–7860, Oct. 2021.

[13] J. Dallas, M. Thompson, J. Y. M. Goh, and A. Balachandran, "Adaptive nonlinear model predictive control: maximizing tire force and obstacle avoidance in autonomous vehicles," *Field Robotics*, vol. 3, pp. 222–242, 2023.

[14] T. Miki *et al.*, "Learning robust perceptive locomotion for quadrupedal robots in the wild," *Science Robotics*, vol. 7, no. 62, p. eabk2822, 2022. [Online]. Available: https://www.science.org/doi/abs/10.1126/scirobotics.abk2822

[15] A. Kumar, Z. Fu, D. Pathak, and J. Malik, "Rma: Rapid motor adaptation for legged robots," in *Robotics: Science and Systems*, 2021.

[16] L. Bauersfeld*, E. Kaufmann*, P. Foehn, S. Sun, and D. Scaramuzza, "NeuroBEM: Hybrid Aerodynamic Quadrotor Model," in *Robotics: Science and Systems XVII*. Robotics: Science and Systems Foundation, Jul. 2021.

[17] Y. Song, A. Romero, M. Müller, V. Koltun, and D. Scaramuzza, "Reaching the limit in autonomous racing: Optimal control versus reinforcement learning," *Science Robotics*, vol. 8, no. 82, p. eadg1462, 2023. [Online]. Available: https://www.science.org/doi/abs/10.1126/scirobotics.adg1462

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

[19] E. Ghignone, N. Baumann, M. Boss, and M. Magno, "TC-Driver: Trajectory Conditioned Driving for Robust Autonomous Racing – A Reinforcement Learning Approach," *Field Robot.*, vol. 3, pp. 637–651, Apr. 2023, arXiv:2205.09370 [cs]. [Online]. Available: http://arxiv.org/abs/2205.09370

[20] M. O'Kelly, H. Zheng, D. Karthik, and R. Mangharam, "F1tenth: An open-source evaluation environment for continuous control and reinforcement learning," in *Proceedings of the NeurIPS 2019 Competition and Demonstration Track*, ser. Proceedings of Machine Learning Research, H. J. Escalante and R. Hadsell, Eds., vol. 123. PMLR, 08–14 Dec 2020, pp. 77–89. [Online]. Available: https://proceedings.mlr.press/v123/o-kelly20a.html

[21] A. Savitzky and M. J. E. Golay, "Smoothing and differentiation of data by simplified least squares procedures." *Analytical Chemistry*, vol. 36, no. 8, pp. 1627–1639, 1964.

[22] A. J. S. I. J.M. Besselink and H. B. Pacejka, "An improved Magic Formula/Swift tyre model that can handle inflation pressure changes," *Vehicle System Dynamics*, vol. 48, no. sup1, pp. 337–352, 2010.

[23] H. B. Pacejka and E. Bakker, "The magic formula tyre model," *Vehicle System Dynamics*, vol. 21, no. sup001, pp. 1–18, 1992. [Online]. Available: https://doi.org/10.1080/00423119208969994

[24] J. Kabzan *et al.*, "AMZ Driverless: The Full Autonomous Racing System," May 2019, arXiv:1905.05150 [cs]. [Online]. Available: http://arxiv.org/abs/1905.05150

[25] J. Węgrzynowski, G. Czechmanowski, P. Kicki, and K. Walas, "Learning dynamics models for velocity estimation in autonomous racing," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024, pp. 972–979.

[26] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=Bkg6RiCqY7

[27] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal Policy Optimization Algorithms," Aug. 2017, arXiv:1707.06347 [cs]. [Online]. Available: http://arxiv.org/abs/1707.06347

[28] A. Hill *et al.*, "Stable baselines," https://github.com/hill-a/stable-baselines, 2018.

[29] F. Djeumou, M. Thompson, M. Suminaka, and J. Subosits, "Reference-free formula drift with reinforcement learning: From driving data to tire energy-inspired, real-world policies." [Online]. Available: http://arxiv.org/abs/2410.20990

[30] J. F. Frenet, "Sur les courbes à double courbure," *Journal de Mathématiques Pures et Appliquées*, 1852.

[31] J. Tobin *et al.*, "Domain randomization for transferring deep neural networks from simulation to the real world," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 23–30.

[32] R. Verschueren *et al.*, "acados – a modular open-source framework for fast embedded optimal control," *Mathematical Programming Computation*, 2021.

[33] R. Verschueren, S. De Bruyne, M. Zanon, J. V. Frasch, and M. Diehl, "Towards time-optimal race car driving using nonlinear MPC in real-time," in *53rd IEEE Conference on Decision and Control*. Los Angeles, CA, USA: IEEE, Dec. 2014, pp. 2505–2510.

[34] T. Novi, A. Liniger, R. Capitani, and C. Annicchiarico, "Real-time control for at-limit handling driving on a predefined path," *Vehicle System Dynamics*, vol. 58, no. 7, pp. 1007–1036, Jul. 2020.

[35] M. Krinner *et al.*, "MPCC++: Model Predictive Contouring Control for Time-Optimal Flight with Safety Constraints," in *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024.