

Drive Fast, Learn Faster: On-Board RL for High Performance Autonomous Racing

Anonymous authors

Paper under double-blind review

Keywords: Autonomous Racing, Physical Robot Learning,

Summary

Autonomous racing presents unique challenges due to its non-linear dynamics, the high speed involved, and the critical need for real-time decision-making under dynamic and unpredictable conditions. Most traditional [Reinforcement Learning \(RL\)](#) approaches rely on extensive simulation-based pre-training, which faces crucial challenges in transfer effectively to real-world environments. This paper introduces a robust on-board [RL](#) framework for autonomous racing, designed to eliminate the dependency on simulation-based pre-training enabling direct real-world adaptation. The proposed system introduces a refined [Soft Actor-Critic \(SAC\)](#) algorithm, leveraging a residual [RL](#) structure to enhance classical controllers in real-time by integrating multi-step [Temporal-Difference \(TD\)](#) learning, an asynchronous training pipeline, and [Heuristic Delayed Reward Adjustment \(HDRA\)](#) to improve sample efficiency and training stability. The framework is validated through extensive experiments on the F1TENTH racing platform, where the residual [RL](#) controller consistently outperforms the baseline controllers and achieves up to an 11.5 % reduction in lap times compared to the [State-of-the-Art \(SotA\)](#) with only 20 min of training. Additionally, an [End-to-End \(E2E\)](#) [RL](#) controller trained without a baseline controller surpasses the previous best results with sustained on-track learning.

Contribution(s)

1. This paper presents an efficient [RL](#) architecture, based on the residual policy structure, that enables uninterrupted on-board learning at an unprecedented level of performance. The architecture is compared to an [E2E](#) structure, showing how the residual policy greatly increases efficiency by simplifying the control task.
Context: The work in ([Stachowicz et al., 2023](#)) presented a high-speed driving solution with on-vehicle data only but failed to compare with [SotA](#) traditional methods for racing, exhibiting a performance gap. We show how a residual structure, previously used by [Trumpp et al. \(2023\)](#) in simulation, can be used to efficiently train in reality only and overtake classical [SotA](#) performance.
2. This paper compares different baseline controllers with the residual [RL](#) policy, showing how the proposed architecture is easily adaptable and improves on all the methods. The fastest agent in the comparison eventually turns out to be the fastest F1TENTH algorithm when compared to previous best results.
Context: The work in ([Trumpp et al., 2023](#)) only used [Pure Pursuit \(PP\)](#) as a baseline controller, while we show that the residual architecture can also adapt to different ones.
3. This paper provides an ablation study in the F1TENTH simulation environment of the different [RL](#) architectural choices, showing how the different parts affect efficiency and training consistency.
Context: Some of the architectural choices were motivated by previous ablation studies (multi-step [TD](#) by [Barth-Maron et al. \(2018\)](#), asynchronous training by [Yuan & Mahmood \(2022\)](#)). Here the full combination is tested in the [Autonomous Racing \(AR\)](#) environment. Furthermore, the introduction of the novel [HDRA](#) is analyzed.
4. This paper provides a generalization study showing how the residual policy behaves with a different type of track and tires in a zero-shot and few-shot setup.
Context: None

Drive Fast, Learn Faster: On-Board RL for High Performance Autonomous Racing

Anonymous authors

Paper under double-blind review

Abstract

Autonomous racing presents unique challenges due to its non-linear dynamics, the high speed involved, and the critical need for real-time decision-making under dynamic and unpredictable conditions. Most traditional [Reinforcement Learning \(RL\)](#) approaches rely on extensive simulation-based pre-training, which faces crucial challenges in transfer effectively to real-world environments. This paper introduces a robust on-board [RL](#) framework for autonomous racing, designed to eliminate the dependency on simulation-based pre-training enabling direct real-world adaptation. The proposed system introduces a refined [Soft Actor-Critic \(SAC\)](#) algorithm, leveraging a residual [RL](#) structure to enhance classical controllers in real-time by integrating multi-step [Temporal-Difference \(TD\)](#) learning, an asynchronous training pipeline, and [Heuristic Delayed Reward Adjustment \(HDRA\)](#) to improve sample efficiency and training stability. The framework is validated through extensive experiments on the F1TENTH racing platform, where the residual [RL](#) controller consistently outperforms the baseline controllers and achieves up to an 11.5 % reduction in lap times compared to the [State-of-the-Art \(SotA\)](#) with only 20 min of training. Additionally, an [End-to-End \(E2E\)](#) [RL](#) controller trained without a baseline controller surpasses the previous best results with sustained on-track learning. These findings position the framework as a robust solution for high-performance autonomous racing and a promising direction for other real-time, dynamic autonomous systems.

1 Introduction

[Reinforcement Learning \(RL\)](#) has become ubiquitous in robotics, particularly on quadrupedal robots, [RL](#) is now almost indispensable for locomotion, as it implicitly learns efficient heuristics for complex movements [Hoeller et al. \(2024\)](#). Similarly, in autonomous drone racing, [RL](#) has exhibited superhuman control capabilities, outperforming traditional approaches and human champions [Kaufmann et al. \(2023\)](#). These successes highlight the necessity of [RL](#) in robotics, particularly for pushing robotic systems to their performance limits.

Despite its success in various robotic domains, [RL](#) has not yet achieved widespread adoption in [Autonomous Racing \(AR\)](#). This is particularly notable, given that [AR](#) also involves agile control of non-linear systems posing significant challenges for robotic platforms [\(Betz et al., 2023\)](#). Moreover, the few solutions found on [AR](#) are showing impressive results only in simulation [\(Fuchs et al., 2021; Vasco et al., 2024\)](#), while evaluations on physical race cars remain limited. One possible explanation is due to the fact that when deployed on real vehicles, [RL](#)-based controllers are consistently slower than [State-of-the-Art \(SotA\)](#) classical [AR](#) controllers [\(Brunnbauer et al., 2022; Ghignone et al., 2025\)](#). This suggests that the [Simulation-to-Reality \(sim-to-real\)](#) gap in [AR](#) is particularly challenging to bridge.

One approach to mitigating this gap involves developing high-fidelity simulators that accurately model vehicle dynamics, adapting them to every possible physical environment. While this ap-

proach has been effective, especially in tandem with data-driven techniques that improve simulation accuracy (Kaufmann et al., 2023), it remains limited by inevitable mismatches between simulated and real-world physics and requires accurate system identification of the physical system and calibration of the simulator (Song et al., 2023). Instead, we take a different approach, bypassing these limitations entirely by training directly on the physical car and allowing the agent to learn and adapt to real-world dynamics from the start.

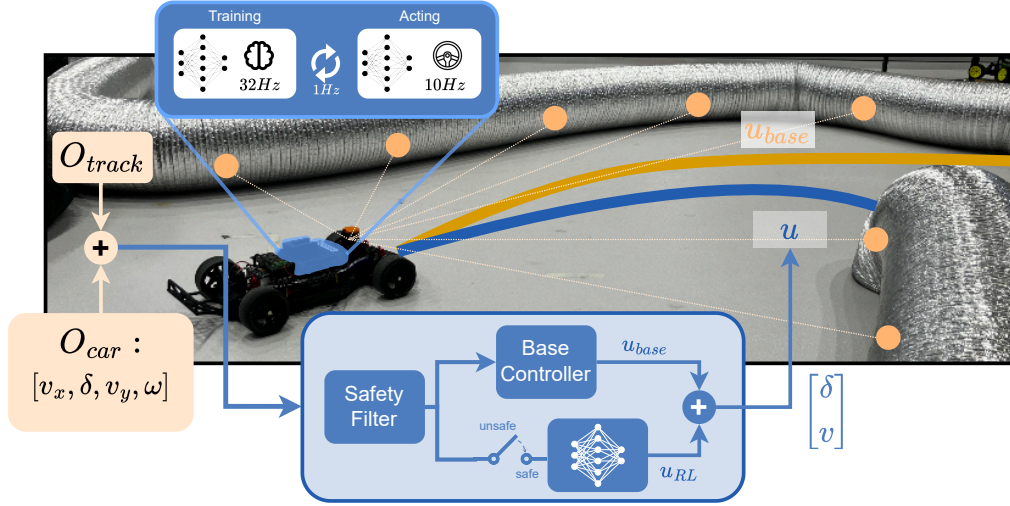


Figure 1: Overview of the proposed RL architecture, which processes observations O_{car} , providing proprioceptive information such as position and velocity (v_x, v_y) , and O_{track} , describing track boundaries and a given reference trajectory to generate a residual control command u_{RL} . Such a command is additively combined with the output of a conventional controller u_{base} . Since the agent starts untrained and may encounter unsafe states, a safety filter is employed to guide the robot back to a safe state. Training happens entirely on-board in an asynchronous fashion: the *Acting* node provides residual control commands at 10 Hz, while the training node updates the control policy at 32 Hz. The policy of the acting node is updated with the policy of the training node every second.

This work proposes an onboard RL-based high-performance AR controller that significantly and consistently outperforms SotA classical controllers in terms of lap time. This work eliminates the sim-to-real gap by training the RL agent (Wu et al., 2023; Stachowicz et al., 2023) directly on the physical robot, namely a 1:10 scaled FITENTH race car (O’Kelly et al., 2020), and validates the approach across multiple track layouts and friction conditions. With accurate real-field measurements the paper demonstrates that an RL agent can surpass a high-performance, model-based SotA racing controller in real-world AR, achieving up to a 11.5 % improvement in lap time, and enabling the physical robot to operate without speed constraints and over 6.5 m s^{-1} . Moreover one important finding out of this approach is that direct on-robot training introduces a new challenge: unlike simulation, where massive parallelization can be leveraged to accelerate learning (Rudin et al., 2021), real-world training remains inherently limited by the time needed for data collection. To address this problem, a residual RL structure is employed (Johannink et al., 2019), enabling a fully untrained agent to learn to exceed the performance of the SotA controller within 20 min. This paper further shows that such architecture significantly accelerates the training process, as an End-to-End (E2E) model (not residual), achieves similar but slower lap times in three times the amount of training.

The proposed training framework is based on the Soft Actor-Critic (SAC) algorithm and incorporates several modifications to facilitate on-board learning. Multi-step Temporal-Difference (TD) updates are utilized to manage delayed rewards (Sutton & Barto, 2018), while Heuristic Delayed Reward Adjustment (HDRA), inspired by Hindsight Experience Replay (HER), is employed to emphasize undesirable behaviors. Additionally, an asynchronous training pipeline is implemented, decoupling data collection from policy optimization, following the approach of Yuan & Mahmood (2022). Fi-

nally, a safety filter allows for uninterrupted data collection, by gating potentially unsafe inputs and rerouting the vehicle to a reference safe trajectory. An overview of the proposed architecture is available in Figure 1.

2 Related Work

RL has seen growing applications in AR due to its potential to handle high-speed decision-making and complex, non-linear vehicle dynamics (Fuchs et al., 2021; Vasco et al., 2024). While this section focuses on previous work that directly informs our approach, Appendix A gives a broader overview of the advancements in autonomous driving and its subfield AR.

The **Sim-to-real gap** has been a persistent issue in applying RL to physical AR systems (Zhao et al., 2020). This gap stems from the inability of simulators to accurately replicate real-world dynamics, sensor noise, latencies, and environmental uncertainties. Methods such as system identification (Sontakke et al., 2023) and domain randomization (Tobin et al., 2017; Peng et al., 2018) have been widely adopted to address this problem. Chisari et al. (2021) proposed an RL framework for miniature race cars that leveraged model randomization, policy regularization, and online fine-tuning of the learned policy to diminish the **sim-to-real** gap, achieving near **Model Predictive Control (MPC)**-level performance. Despite these advances, **sim-to-real** methods often require extensive pre-training in simulation and substantial fine-tuning on physical systems, which can be resource-intensive and time-consuming and do not reach the performance of classical **SotA** controllers. Our proposed architecture bypasses these problems entirely by learning directly on-board.

Residual RL (Johannink et al., 2019) proposes a structure in which an agent learns a residual policy that adjusts or refines the actions of a pre-existing base controller, allowing for improved performance and adaptability. Trumpp et al. (2023) introduced a residual policy learning approach specifically for AR in simulation. Their method involved training the residual policy to adjust a base **Pure Pursuit (PP)** controller’s actions and demonstrated to be effective in high-speed racing tasks by reducing lap times and enhancing stability when evaluated on simulated F1TENTH racing tracks. Ghignone et al. (2025) deployed a similar approach on the physical F1TENTH platform. Such an agent, trained in simulation, was then able to improve the lap time of the base controller with the help of an RL policy. However, the achieved lap times were still significantly slower than **SotA** classical algorithms.

In this work, the residual RL approach is adopted due to its faster convergence to fast lap times, essential for achieving on-board learning with limited time.

On-Robot learning eliminates the **sim-to-real** gap entirely by training directly on physical robots but needs particularly efficient structures to achieve high performance (Wu et al., 2023), as the massively parallel architectures that are available in simulation (Rudin et al., 2021) are often not replicable. Some works have been developed directly on-board, such as in Bosello et al. (2022), where an agent could successfully navigate an oval track with three hours of training, relying solely on **Light Detection And Ranging (LiDAR)** scans for its observations. The final policy achieved speeds of approximately 2 m/s. Similarly, Evans et al. (2023b) trained an agent under the supervision of a safety system based on a viability kernel designed to ensure crash-free operation. With this setup, the agent outperformed a simulation-trained baseline in terms of lap times and reliability, achieving efficient training within 10 min. However, the approach was constrained by a speed limit of 2 m/s and required the vehicle to pause every 20 steps to update the policy, highlighting areas for improvement in achieving continuous, high-speed training. FastRLAP (Stachowicz et al., 2023) introduced an online RL framework for AR, achieving stable driving behavior within 20 min of real-world training with an image-to-action approach. However, the approach required an external workstation for policy updates, constraining its scalability and applicability in resource-constrained environments, and was limited to speeds up to 4.5 m/s. Pan et al. (2020) used imitation learning on a scaled car to match an MPC expert used to generate data. However, this approach did not surpass the **SotA** performance and depended on image-to-action methods and an onboard **Graphical Processing Unit (GPU)**.

In contrast to the previous work, our contribution consistently surpasses the [SotA AR](#) agents, both with and without the residual structure, and displays agile racing, without any speed limitation and reaching over 6.5 m s^{-1} . Similarly to other algorithms, our algorithm needs around 20 min of training, but it only trains locally (on the scaled autonomous platform) and with no access to a [GPU](#). Finally, our proposed solution is validated on a real platform across multiple track configurations and tire conditions, highlighting the empirical robustness of the method.

3 Methodology

The residual control architecture utilized throughout this work is illustrated in Figure 1. Detailed descriptions of each block are given in the subsequent sections.

This study is conducted and validated on the F1TENTH platform ([O’Kelly et al., 2020](#)), a scaled racing platform with form factor 1:10. Localization and state estimation on a mapped track layout is based on a [Simultaneous Localization and Mapping \(SLAM\)](#) system using a 2D [LiDAR](#) sensor and [Inertial Measurement Unit \(IMU\)](#). The hardware setup, software architecture (implemented using the [Robot Operating System \(ROS\)](#)), and the localization pipeline closely replicate the approach outlined in [Baumann et al. \(2024\)](#). All computations are conducted on the onboard Intel i7-10710U processor.

3.1 State Machine

To facilitate efficient and autonomous training without human intervention, a state machine is implemented to recover the vehicle from terminal states. Two terminal states are defined: the first occurs when a track-boundary violation is detected, which happens if the center of the rear axle moves outside the defined track boundaries. The second terminal state is triggered by the intervention of the relative-heading safety filter, as detailed in the subsequent section.

Upon reaching a terminal state, the [RL](#) agent is deactivated and the baseline controller steers the car back to the reference trajectory. Once the vehicle orientation realigns with the reference trajectory, a new learning episode is initiated, and the residual controller resumes operation. This process ensures seamless episode transitions and continuous data collection, enabling stable and efficient training.

3.2 Curriculum Safety Filter

A relative-heading safety filter is implemented to prevent the vehicle from leaving the track with a large relative heading, $\Delta\psi$, with respect to the reference trajectory. The safety filter signals a terminal state if $|\Delta\psi| > \psi_{\text{filter}}$, where $\Delta\psi = \psi - \psi_{\text{ref}}$, ψ is the vehicle’s current heading, and ψ_{ref} is the heading of the reference trajectory in the global frame. The safety filter is implemented with an adaptive threshold, ψ_{filter} . If the agent completes a lap successfully, ψ_{filter} is increased by ϵ ; conversely, it is decreased by the same amount if a track-boundary violation is detected.

This mechanism serves two purposes: first, it prevents the vehicle from traversing the run-off area with a heading towards the physical barriers, giving the state machine more time to intervene. Second, it facilitates the agent’s learning process by acting as a curriculum mechanism ([Bengio et al., 2009](#)), forcing it to follow the reference trajectory more closely in the beginning, before relaxing the constraint.

3.3 Action Space

The control command is defined as $u = [\delta, v]^T$, where δ is the steering angle, and v the velocity. The command is the sum of the base controller’s output, $u_{\text{base}} = [\delta_{\text{base}}, v_{\text{base}}]^T$, and the [RL](#) controller’s output, $u_{\text{RL}} = [\delta_{\text{RL}}, v_{\text{RL}}]^T$.

The base controller operates within the ranges $\delta_{\text{base}} \in [-0.42, 0.42] \text{ rad}$ and $v_{\text{base}} \in [0, 10] \text{ m/s}$. The learned policy refines u_{base} with residual adjustments, constrained to $\delta_{\text{RL}} \in [-0.15, 0.15] \text{ rad}$ and

159 $v_{RL} \in [-0.5, 2] \text{ m/s}$. This formulation allows the residual controller to make fine adjustments to the
 160 base controller’s output, enabling greater adaptability, precision, and ultimately faster lap times.

161 3.4 Observation Space

162 All observations are based on the onboard state estimation. The observation space
 163 is constructed similarly to Ghignone et al. (2024), incorporating vehicle state, dynam-
 164 ics, and track information. The vehicle dynamics and state are represented by $o_{car} =$
 165 $[v_x, v_y, \dot{\psi}, \Delta d, \Delta\psi, \delta_{base}, v_{base}, \delta_{RL, prev}, v_{RL, prev}]^T$, where v_x and v_y are the lateral and longitudinal
 166 velocities in the car frame, $\dot{\psi}$ is the yaw rate, Δd is the car’s lateral deviation from the reference
 167 line, and $\Delta\psi$ is its relative heading to the reference line. δ_{base} and v_{base} are the outputs of the base
 168 controller, and $\delta_{RL, prev}$ and $v_{RL, prev}$ are the outputs of the RL controller of the previous time step.

169 To capture track information, the observation space includes o_{track} , which summarizes the upcoming
 170 reference trajectory and track boundaries. A total of J equally spaced points along the reference
 171 trajectory are sampled within a horizon of l meters. At each sampled point, the positions of the
 172 reference line, left boundary, and right boundary are stored in the car frame. Formally, this is
 173 expressed as $o_{track} = [\mathbf{p}_{ref}^0, \dots, \mathbf{p}_{ref}^J, \mathbf{p}_{left}^0, \dots, \mathbf{p}_{left}^J, \mathbf{p}_{right}^0, \dots, \mathbf{p}_{right}^J]^T$, where $\mathbf{p}_{ref} \in \mathbb{R}^2$ represents
 174 a two-dimensional point on the race line, and \mathbf{p}_{left} and \mathbf{p}_{right} represent points on the left and right
 175 track boundaries, respectively. The complete observation space is then defined as $o = [o_{car}, o_{track}]^T$,
 176 with $o \in \mathbb{R}^{5+6J}$ and is normalized. A graphical representation of the observation space is available
 177 in the supplementary material, Figure 7.

178 3.5 Reward Function

179 The reward consists of a positive term $\lambda\Delta s$, granted at every step for progress along the reference
 180 line Δs and scaled by the constant λ (Chisari et al., 2021; Ghignone et al., 2024), as well as a
 181 negative penalty p , applied only upon track-boundary violations or safety filter interventions. Both
 182 terms are mutually exclusive.

183 3.6 Training Architecture

184 To improve data efficiency, we extend SAC (Haarnoja et al., 2018) from Stable Baselines (Hill
 185 et al., 2018) with an asynchronous pipeline that separates the agent into an acting policy actor and a
 186 training policy, akin to *Async-SAC-I* from Yuan & Mahmood (2022). Then the proposed approach
 187 incorporates multi-step TD learning (Sutton & Barto, 2018), similarly to previous AR work (Vasco
 188 et al., 2024), to stabilize training (Barth-Maron et al., 2018).

189 Additionally, a HDRA mechanism, inspired by principles from HER (Andrychowicz et al., 2017)
 190 and reward shaping (Ng et al., 1999), has been adopted and integrated. In scenarios where the agent
 191 reaches a terminal state with a negative reward, the N most recent transitions in the replay buffer
 192 are retroactively modified to amplify the penalty and create a stronger learning signal to discourage
 193 failure and the actions leading up to it. The stored rewards are then adjusted as follows: given an
 194 index $i_{term} \in \mathbb{N}$ corresponding to a terminal state stored in the replay buffer, the reward associated
 195 with the action that led to this state is $r_{i_{term}-1} = p$. The N previous rewards (r^{old}) are then
 196 decreased with a linearly decreasing term as follows:

$$r_{i_{term}-1-j} = r_{i_{term}-1-j}^{old} - \frac{N-j}{N}p, \quad \forall j \in [0, 1, \dots, N-1]. \quad (1)$$

197 The corresponding pseudocode is attached in Appendix G.

198 During training, the control frequency of the residual RL controller is set to 10 Hz, with the training
 199 policy updated at 32 Hz and the behavior policy synced with the latter at 1 Hz. During deployment,
 200 the control frequency is increased to 15 Hz, while the baseline controllers operate at a fixed fre-
 201 quency of 40 Hz throughout both training and deployment phases. The output of the RL controller
 202 is continuously added to the output of the base controller.

4 Experimental Setup

The effectiveness and robustness of the proposed architecture are evaluated through a series of time-trial experiments. The primary objective is to minimize lap time while maintaining stability and reliability over multiple laps. Each scenario is allocated a single 5000 mA h battery for training, which lasts between 20 and 33 min depending on the track-tire configuration and speed profile. After training, the agent is deployed until it completes 20 laps without track-boundary violations. Metrics such as lap times and number of boundary violations are recorded to evaluate speed, repeatability, and adaptability.

4.1 Evaluation

One of the most important goals of this evaluation is to demonstrate the generalization of the proposed approach. This is done by integrating the residual **RL** controller with three distinct baseline controllers (**PP**, **Model and Acceleration-Based Pursuit (MAP)**, and **Follow the Gap (FTG)**, as detailed in Section 4.2) and by testing it in an **E2E** configuration. Each configuration was evaluated on the 41 m long **C-track**, with performance measured by the minimum lap time over 20 consecutive deployment laps.

Following these initial experiments, the best-performing combination was selected for further evaluation. These extended tests included repeated training and deployment with varying random seeds on the **C-track** and on the 34 m long **Y-track** to assess repeatability and robustness. While these primary experiments were conducted using Turbo tires with a relatively high static friction coefficient of $\mu_{static}^{Turbo} \approx 1.01$, additional experiments were performed on the **C-track** using **Thermoplastic Polyurethane (TPU)** tires, which exhibit a lower static friction coefficient of $\mu_{static}^{TPU} \approx 0.75$ and different cornering behavior. This allowed for a comprehensive analysis of the controller’s performance under varying track and tire conditions. Images of the **F1TENTH** car, tires, and track setup can be seen in Appendix B.

4.2 Baselines

The following baseline controllers are used to evaluate the agent’s learning enhancements and integration capabilities with different baseline controllers:

PP: A classical geometric controller that calculates steering based on a look-ahead point on a pre-defined reference trajectory. This controller operates under a no-slip assumption, making it limited at high speeds where friction dynamics become significant (Coulter, 1992).

Model and Acceleration-Based Pursuit (MAP): An extended pursuit method that integrates a lookup table to incorporate tire dynamics, allowing for improved handling in high-speed corners by accounting for non-linear friction effects (Becker et al., 2023).

Follow the Gap (FTG): This reactive approach identifies gaps in the **LiDAR** measurements to navigate through (Sezer & Gokasan, 2012). Therefore, it is often the chosen controller when navigating unknown environments with physical boundaries. However, due to its reactive behavior and lack of knowledge regarding the upcoming track sections, it often lags behind in terms of lap time.

End-to-End (E2E): To demonstrate the data efficiency of the proposed **RL** architecture and isolate its performance, experiments without a baseline controller are conducted. The range for the commanded steering angle is extended to $[-0.42; 0.42]$ rad, which aligns with the limits of the physical steering actuator. For the commanded speed, a curriculum learning approach has been implemented (Wang et al., 2022). The action space for the speed command is $[0.5; \alpha]$ with $\alpha \in [1; 7]$ m/s. α is increased by 0.5 m/s as soon as the agent completes three consecutive laps without track-boundary violation. Naturally, the baseline control command is removed from the observation space.

Remark: While **MPC** provides precise trajectory tracking and constraint adherence, it is not combined with the residual controller in this study as its constraint-based nature inherently conflicts with

RL’s adaptive learning, especially when the agent attempts to operate near the limits of the MPC’s predefined constraints. Nonetheless, we include MPC performance in our baseline evaluation, as it is recognized as one of the most performant controllers in the literature (Vázquez et al., 2020) and provides a valuable reference point for contextualizing our results. More detail on the optimization problem formulation, also derived from the work of Vázquez et al. (2020), is available in Appendix J.

5 Results

Figure 2 shows the lap time improvement during training for the different residual RL controllers and the E2E RL controller on the C-track, highlighting the agent’s ability to improve upon the performance of all baseline controllers. While RL MAP and RL PP outperformed their respective base controller within 10 min, RL FTG needed about 20 min to achieve the same. The E2E agent also shows significant lap time improvements within 20 min, but does not manage to reach the SotA baseline set by the MAP controller.

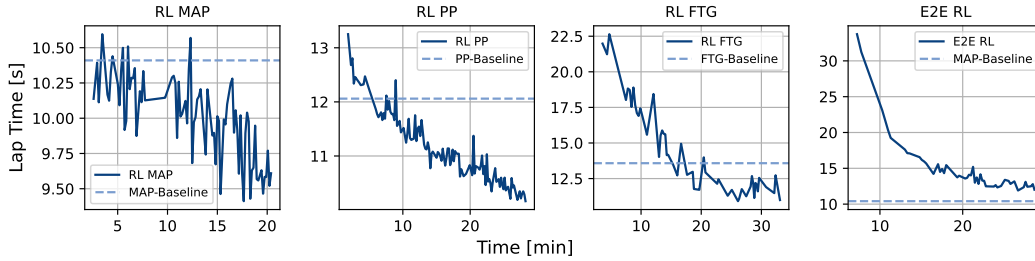


Figure 2: Lap Time vs. Wall Time during training for different residual controllers, with respective reference for best lap time of the baseline controller and the E2E controller with MAP baseline. Only lap times without boundary violations are recorded.

During deployment, the RL MAP controller achieved the fastest lap time of 9.26 s, significantly surpassing the MAP baseline’s best of 10.41 s. Across all baseline-residual RL combinations, lap time reductions ranged from 11.0 % to 20.5 % for best lap times and 9.0 % to 20.3 % for mean lap times. An extended table listing all minimum, average and the Standard Deviation (SD) of the lap times for the different combinations can be seen in Appendix C. RL MAP’s superior results in terms of minimum and average lap time made it the primary choice for subsequent experiments.

E2E extended training: Since training the E2E controller for only one battery life was not sufficient to obtain SotA performance, the experiment was extended to three battery lives, cumulating in 82 min of on-track learning. Afterward, the agent’s average lap time was 5.0 % lower than the SotA set by the MAP controller, while the minimum lap time was 7.3 % lower (Extended E2E RL: $t_{min} = 9.65$ s, $t_{\mu} = 9.97$ s. MAP: $t_{min} = 10.41$ s, $t_{\mu} = 10.50$ s). During the deployment, only 4 track-bound violations and speeds up to 7 m s^{-1} were registered. Extended data and visualizations of the deployment runs can be seen in Appendix C.

5.1 RL MAP Extended Evaluation

The RL MAP controller was further re-trained from scratch in two different conditions and across three different random seeds to evaluate the method across environments. The first environment consisted of using the more slippery TPU tires on the same track (C-track), while the second experiment evaluated the algorithm on the Y-track. Tires pictures and track layouts are available in Appendix B and Appendix E. The RL MAP controller consistently outperformed its baseline across all scenarios tested, achieving improvements ranging from 5.4% to 11.5% for minimum lap times and 5.9% to 9.0% for mean lap times. Even during the worst deployment run in terms of

lap time reduction, the proposed architecture managed to decrease the minimum lap time by 0.57 s and the average lap time by 0.51 s. This highlights the robustness of the approach across varying track layouts, friction levels, and weight initialization of the neural network. Table 1 summarizes the three deployment runs per tire and track combination. A detailed break-down of the individual deployments is presented in the Appendix E.

Ctrl.	C-Track, Turbo				C-Track, TPU				Y-Track, Turbo			
	$t_{\min} \downarrow$	$t_{\mu} \downarrow$	$\sigma \downarrow$	$n_b \downarrow$	$t_{\min} \downarrow$	$t_{\mu} \downarrow$	$\sigma \downarrow$	$n_b \downarrow$	$t_{\min} \downarrow$	$t_{\mu} \downarrow$	$\sigma \downarrow$	$n_b \downarrow$
MPC	11.86	11.95	0.083	0	13.14	13.25	0.179	0	9.97	10.04	0.071	1
MAP	10.41	10.50	0.060	0	12.93	13.45	0.338	0	8.55	8.62	0.045	0
RL MAP (Ours)	9.26	9.56	0.181	2	12.23	12.50	0.163	1	7.57	7.97	0.205	6

Table 1: Performance of RL MAP with different tire-track combinations averaged over three randomly initialized seeds compared to MAP and MPC for reference. t_{\min} [s] represents the minimum lap time, t_{μ} [s] the average lap time, σ [s] the standard deviation of the lap times and n_{bound} the number of boundary violations until 20 violation-free laps are reached.

In terms of lap time repeatability, RL MAP exhibited a more than three times larger lap time standard deviation compared to its baseline controller on the high-friction Turbo tires. However, in the low-friction scenario, RL MAP demonstrated greater consistency with an SD of 0.16 s, producing more stable lap times than its baseline controller with a SD of 0.34 s. While the MAP baseline was capable of completing 20 consecutive laps without any track-boundary violations across all scenarios, the residual agent experienced up to two violations on the C-track with Turbo tires, up to one violation on the C-track with TPU tires, and up to five on the Y-track track before successfully completing 20 violation-free laps. The majority of these violations occurred due to slight corner clipping.

Qualitative observations indicate that the RL MAP controller adjusts steering and speed effectively across the track. For example, on the C-track with Turbo tires (Figure 3) the residual controller negotiates all the corners but one with higher minimum speed, without sacrificing straight speed, as it also reaches higher top velocities at the end of the acceleration sections. Similar behavior can be observed in the other configurations in the supplementary material, Appendix E.

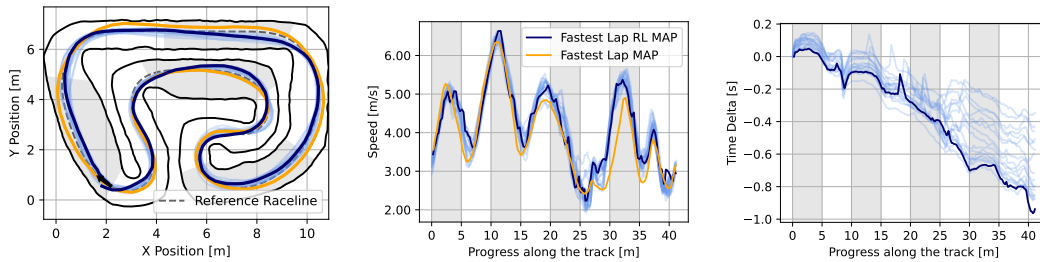


Figure 3: Comparison of the driven trajectories, speed profiles, and the time delta for RL MAP (blue) vs. MAP (orange) on the C-track with Turbo tires. Speed profiles and time deltas are represented along the positional advancement on the track. The zero progress position corresponds to the black arrow in the bottom left corner, and the track is driven in the sense of the arrow. Dark blue represents the fastest lap of RL MAP.

5.2 RL MAP Generalization Evaluation

Generalization to a different track layout was evaluated by deploying an RL MAP agent, trained on the C-track with Turbo tires, on the Y-track with same tires using zero-shot (no additional training) and few-shot transfer (additional training for one battery life on the new track).

In zero-shot transfer, the results across random seeds were mixed and not all 20 laps could be completed, with the aggregate deployment attaining $t_\mu = 8.68 s$, $\sigma = 0.34 s$. While Seed 1 managed to outperform the baseline in terms of best and mean lap times after only one collision, Seed 2 and Seed 3 experienced significant instability, with multiple track-boundary violations that necessitated terminating the deployment.

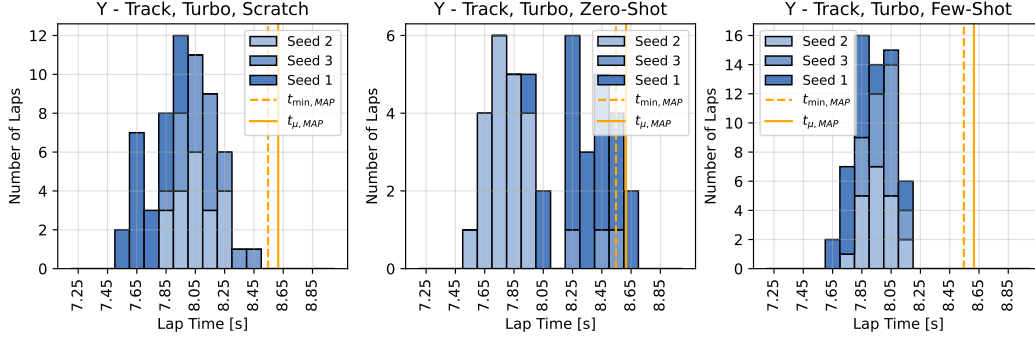


Figure 4: Stacked histogram of lap times achieved during zero-shot and few-shot transfer from the C-track to Y-track during 20 boundary-violation-free laps. Different shades represent the three different weight initialization and resulting policies learned on the C-track.

Due to the very limited zero-shot transfer capabilities of the method, we proceeded to further test few-shot transfer, which involved additional training for one battery life. This led to improvements in both lap repeatability and lap times compared to zero-shot transfer. When compared to policies trained from scratch on the Y-track, the few-shot transfer policy demonstrated fewer track-boundary violations (up to two per seed, four in total), an on average 4.7 % lower mean lap time and half the standard deviation in lap times across all random seeds. The aggregate laps demonstrated $t_\mu = 7.93 s$, $\sigma = 0.13 s$. Figure 4 presents a distribution of all the lap times recorded for the evaluation, highlighting the effectiveness of few-shot transfer in enhancing lap repeatability. Extended results of the generalization tests can be found in Appendix F.

5.3 Ablation Study

We evaluate the impact of key architectural and algorithmic modifications with an ablation study conducted in the real-time F1TENTH ROS simulator (O’Kelly et al., 2019). This study focuses on the more challenging E2E setting, comparing various combinations of multi-step TD learning, HDRA, and training architectures (asynchronous vs. synchronous). Figure 5 presents the results, averaged over five independent training runs with different random seed initializations.

A first comparison shows that in our experiments the asynchronous architecture (HDRA + TD3, Async) manages to reduce the lap times faster than the synchronous architecture (HDRA + TD3, Sync), and in the same training time achieves a faster lap time compared to its synchronous counterpart. This improvement

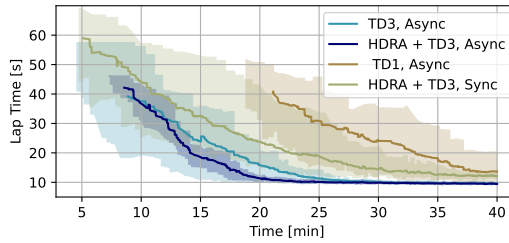


Figure 5: Lap times during training averaged over 5 runs for ablation regarding multi-step TD over one (TD1) and three (TD3) steps, a synchronous (sync) and asynchronous (async) architecture as well as with HDRA and without. Lines only start when the first collision-free lap is completed. The area between the minimum and the maximum lap time is shaded.

is likely due to the ability of the asynchronous architecture to support a policy update rate three times higher.

Furthermore, the addition of multi-step TD error (TD3, Async), shows in our experiments to help speed up the learning process, achieving lap completions earlier than the ten-minute mark, while (TD1, Async) needs in this case around 20 min. Finally, the addition of HDRA (HDRA + TD3, Async) does not show a statistically relevant improvement against (TD3, Async) in average lap time, but keeps the lap time spread lower, and makes the training more consistent across random seeds. Detailed metrics on the lap times achieved during the last three minutes of training are available in Appendix H.

6 Conclusion

This work presented a real-world on-board learning framework for AR, applied to a residual and E2E control framework. The residual controller consistently improves the performance of its baseline across all tested scenarios, achieving substantial reductions in minimum and average lap times. Among the tested configurations, the RL MAP controller delivers the fastest lap times, improving the minimum lap time by up to 11.5 % and the average lap time by up to 10.0 % compared to the classical MAP controller. Beyond improved performance, it demonstrates strong repeatability and stability across different tracks and tire configurations, as well as weight initializations. Moreover, the proposed asynchronous framework is highly efficient, reaching said performance within a single battery life of 20 to 33 min (10 k to 19 k environmental steps) for training.

The E2E RL controller further illustrates the versatility of the architecture. After 29 min of training, it navigated the track consistently at speeds up to 4 m s^{-1} and with 82 min of training, surpassed the MAP baseline by over half a second, maintaining stable performance at speeds up to 7 m s^{-1} . The ablation study shows that especially the asynchronous structure and multi-step TD learning attributes to the fast convergence to the minimal lap time.

Furthermore, while the few-shot generalization experiments show that a single policy can be tuned to drive consistently on a different track, the zero-shot experiments show that future work could focus on methods targeting generalization across different real-world conditions. For example, recent work on data-driven simulation (Xiao et al., 2024; Rothfuss et al., 2024) could be a starting point for creating a generalist policy that would then benefit from the framework proposed here for on-board fine-tuning.

Overall, these findings highlight the potential of a residual RL framework to enhance classical controllers, while the same framework enables E2E RL agents to achieve competitive real-world performance. Future research could focus on reducing track-boundary violations during training via a model-based safety filter (Evans et al., 2023a; Tearle et al., 2021) or extending the residual control architecture to multi-agent racing, which would introduce new challenges, such as dynamic collision avoidance and strategic adaptation. Finally, the proposed architecture could be adapted for full-scale vehicles.

Broader Impact Statement

The versatility of the framework also makes it a candidate for other domains, including drone racing, robotic locomotion, and manipulation tasks. In future work, we are considering exploring how these applications could validate this work’s effectiveness in different high-speed or precision-critical scenarios, expanding its relevance to a broader range of autonomous systems.

References

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight Experience Re-

- 384 play. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.,
385 2017.
- 386 Gabriel Barth-Maron, Matthew W. Hoffman, David Budden, Will Dabney, Dan Horgan, Dhruva
387 TB, Alistair Muldal, Nicolas Heess, and Timothy Lillicrap. Distributional policy gradients. In
388 *International Conference on Learning Representations*, 2018. URL [https://openreview.](https://openreview.net/forum?id=SyZipzbCb)
389 [net/forum?id=SyZipzbCb](https://openreview.net/forum?id=SyZipzbCb).
- 390 Nicolas Baumann, Edoardo Ghignone, Jonas Kühne, Niklas Bastuck, Jonathan Becker, Nadine Im-
391 holz, Tobias Kränzlin, Tian Yi Lim, Michael Lötscher, Luca Schwarzenbach, Luca Tognoni,
392 Christian Vogt, Andrea Carron, and Michele Magno. ForzaETH Race Stack – Scaled Autonomous
393 Head-to-Head Racing on Fully Commercial off-the-Shelf Hardware. *Journal of Field Robotics*, 3
394 2024. ISSN 15564967. DOI: 10.1002/rob.22429.
- 395 Jonathan Becker, Nadine Imholz, Luca Schwarzenbach, Edoardo Ghignone, Nicolas Baumann,
396 and Michele Magno. Model- and Acceleration-based Pursuit Controller for High-Performance
397 Autonomous Racing. In *2023 IEEE International Conference on Robotics and Automation*
398 *(ICRA)*, pp. 5276–5283, 2023. DOI: 10.1109/ICRA48891.2023.10161472.
- 399 Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In
400 *Proceedings of the 26th annual international conference on machine learning*, pp. 41–48, 2009.
- 401 Johannes Betz, Tobias Betz, Felix Fent, Maximilian Geisslinger, Alexander Heilmeier, Leonhard
402 Hermansdorfer, Thomas Herrmann, Sebastian Huch, Phillip Karle, Markus Lienkamp, Boris
403 Lohmann, Felix Nobis, Levent Ögretmen, Matthias Rowold, Florian Sauerbeck, Tim Stahl,
404 Rainer Trauth, Frederik Werner, and Alexander Wischnewski. Tum autonomous motorsport: An
405 autonomous racing software for the indy autonomous challenge. *Journal of Field Robotics*, 40(4):
406 783–809, 2023. DOI: <https://doi.org/10.1002/rob.22153>. URL [https://onlinelibrary.](https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.22153)
407 [wiley.com/doi/abs/10.1002/rob.22153](https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.22153).
- 408 Michael Bosello, Rita Tse, and Giovanni Pau. Train in Austria, Race in Montecarlo: Gen-
409 eralized RL for Cross-Track F1TENTH LIDAR-Based Races. In *2022 IEEE 19th Annual*
410 *Consumer Communications & Networking Conference (CCNC)*, pp. 290–298, 2022. DOI:
411 10.1109/CCNC49033.2022.9700730.
- 412 Alberto Broggi, Massimo Bertozzi, Alessandra Fascioli, and Gianni Conte. *Automatic Vehicle Guid-*
413 *ance*. WORLD SCIENTIFIC, 1999. DOI: 10.1142/3986.
- 414 Axel Brunnbauer, Luigi Berducci, Andreas Brandstätter, Mathias Lechner, Ramin Hasani, Daniela
415 Rus, and Radu Grosu. Latent Imagination Facilitates Zero-Shot Transfer in Autonomous Racing.
416 In *2022 International Conference on Robotics and Automation (ICRA)*, pp. 7513–7520, 2022.
417 DOI: 10.1109/ICRA46639.2022.9811650.
- 418 Eugenio Chisari, Alexander Liniger, Alisa Rupenyan, Luc Van Gool, and John Lygeros. Learn-
419 ing from simulation, racing in reality. In *2021 IEEE International Conference on Robotics and*
420 *Automation (ICRA)*, pp. 8046–8052. IEEE, 2021.
- 421 RC Coulter. Implementation of the Pure Pursuit Path Tracking Algorithm. *DTIC Document*, 1992.
- 422 Marc Peter Deisenroth and Carl Edward Rasmussen. Pilco: a model-based and data-efficient ap-
423 proach to policy search. In *Proceedings of the 28th International Conference on International*
424 *Conference on Machine Learning*, ICML’11, pp. 465–472, Madison, WI, USA, 2011. Omnipress.
425 ISBN 9781450306195.
- 426 Onur Dikici, Edoardo Ghignone, Cheng Hu, Nicolas Baumann, Lei Xie, Andrea Carron, Michele
427 Magno, and Matteo Corno. Learning-based on-track system identification for scaled autonomous
428 racing in under a minute. *IEEE Robotics and Automation Letters*, 10(2):1984–1991, 2025. DOI:
429 10.1109/LRA.2025.3527336.

- 430 Benjamin D. Evans, Hendrik W. Jordaan, and Herman A. Engelbrecht. Safe reinforcement learning
431 for high-speed autonomous racing. *Cognitive Robotics*, 3:107–126, 2023a. ISSN 2667-2413.
432 DOI: <https://doi.org/10.1016/j.cogr.2023.04.002>.
- 433 Benjamin David Evans, Johannes Betz, Hongrui Zheng, Herman A. Engelbrecht, Rahul
434 Mangharam, and Hendrik W. Jordaan. Bypassing the Simulation-to-Reality Gap: Online Re-
435 inforcement Learning Using a Supervisor. In *2023 21st International Conference on Advanced
436 Robotics (ICAR)*, pp. 325–331, 2023b. DOI: 10.1109/ICAR58858.2023.10406465.
- 437 Benjamin David Evans, Herman Arnold Engelbrecht, and Hendrik Willem Jordaan. High-Speed
438 Autonomous Racing Using Trajectory-Aided Deep Reinforcement Learning. *IEEE Robotics and
439 Automation Letters*, 8(9):5353–5359, 2023c. DOI: 10.1109/LRA.2023.3295252.
- 440 Benjamin David Evans, Hendrik Willem Jordaan, and Herman Arnold Engelbrecht. Comparing deep
441 reinforcement learning architectures for autonomous racing. *Machine Learning with Applications*,
442 14:100496, 2023d. ISSN 2666-8270. DOI: <https://doi.org/10.1016/j.mlwa.2023.100496>.
- 443 Florian Fuchs, Yunlong Song, Elia Kaufmann, Davide Scaramuzza, and Peter Durr. Super-Human
444 Performance in Gran Turismo Sport Using Deep Reinforcement Learning. *IEEE Robotics and
445 Automation Letters*, 6:4257–4264, 7 2021. ISSN 23773766. DOI: 10.1109/LRA.2021.3064284.
- 446 Edoardo Ghignone, Nicolas Baumann, and Michele Magno. Tc-driver: A trajectory-conditioned
447 reinforcement learning approach to zero-shot autonomous racing. *IEEE Transactions on Field
448 Robotics*, 1:527–536, 2024. DOI: 10.1109/TFR.2024.3499912.
- 449 Edoardo Ghignone, Nicolas Baumann, Cheng Hu, Jonathan Wang, Lei Xie, Andrea Carron, and
450 Michele Magno. Rlpp: A residual method for zero-shot real-world autonomous racing on scaled
451 platforms, 2025. URL <https://arxiv.org/abs/2501.17311>.
- 452 Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Region-Based Convolutional Net-
453 works for Accurate Object Detection and Segmentation. *IEEE Transactions on Pattern Analysis
454 and Machine Intelligence*, 38(1):142–158, 2016. DOI: 10.1109/TPAMI.2015.2437384.
- 455 Sorin Grigorescu, Bogdan Trasnea, Tiberiu Cocias, and Gigel Macesanu. A survey of deep learning
456 techniques for autonomous driving. *Journal of Field Robotics*, 37(3):362–386, 2020. DOI: <https://doi.org/10.1002/rob.21918>.
- 458 Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft Actor-Critic: Off-Policy
459 Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In Jennifer Dy and
460 Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*,
461 volume 80 of *Proceedings of Machine Learning Research*, pp. 1861–1870. PMLR, 10–15 Jul
462 2018.
- 463 Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy P. Lillicrap. Mastering diverse domains
464 through world models. *CoRR*, abs/2301.04104, 2023. URL [https://doi.org/10.48550/
465 arXiv.2301.04104](https://doi.org/10.48550/arXiv.2301.04104).
- 466 Alexander Heilmeyer, Alexander Wischnewski, Leonhard Hermansdorfer, Johannes Betz, Markus
467 Lienkamp, and Boris Lohmann. Minimum curvature trajectory planning and control for an
468 autonomous race car. *Vehicle System Dynamics*, 58(10):1497–1527, 10 2020. ISSN 0042-3114.
469 DOI: 10.1080/00423114.2019.1631455.
- 470 Ashley Hill, Antonin Raffin, Maximilian Ernestus, Adam Gleave, Anssi Kanervisto, Rene Traore,
471 Prafulla Dhariwal, Christopher Hesse, Oleg Klimov, Alex Nichol, Matthias Plappert, Alec Rad-
472 ford, John Schulman, Szymon Sidor, and Yuhuai Wu. Stable Baselines. [https://github.
473 com/hill-a/stable-baselines](https://github.com/hill-a/stable-baselines), 2018.

- David Hoeller, Nikita Rudin, Dhionis Sako, and Marco Hutter. Anymal parkour: Learning agile navigation for quadrupedal robots. *Science Robotics*, 9(88):eadi7566, 2024. DOI: 10.1126/scirobotics.adi7566. URL <https://www.science.org/doi/abs/10.1126/scirobotics.adi7566>.
- Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/5faf461eff3099671ad63c6f3f094f7f-Paper.pdf.
- Tobias Johannink, Shikhar Bahl, Ashvin Nair, Jianlan Luo, Avinash Kumar, Matthias Loskyll, Juan Aparicio Ojea, Eugen Solowjow, and Sergey Levine. Residual reinforcement learning for robot control. In *2019 international conference on robotics and automation (ICRA)*, pp. 6023–6029. IEEE, 2019.
- Elia Kaufmann, Leonard Bauersfeld, Antonio Loquercio, Matthias Müller, Vladlen Koltun, and Davide Scaramuzza. Champion-level drone racing using deep reinforcement learning. *Nature*, 620(7976):982–987, 2023.
- Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. PointPillars: Fast Encoders for Object Detection From Point Clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
- Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: Single Shot MultiBox Detector. In *Computer Vision – ECCV 2016*, pp. 21–37, Cham, 2016. Springer International Publishing. ISBN 978-3-319-46448-0.
- Andrew Y. Ng, Daishi Harada, and Stuart J. Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the Sixteenth International Conference on Machine Learning, ICML '99*, pp. 278–287, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc. ISBN 1558606122.
- Matthew O’Kelly, Varundev Sukhil, Houssam Abbas, Jack Harkins, Chris Kao, Yash Vardhan Pant, Rahul Mangharam, Dipshil Agarwal, Madhur Behl, Paolo Burgio, and Marko Bertogna. F1/10: An Open-Source Autonomous Cyber-Physical Platform. *ArXiv*, abs/1901.08567, 2019. URL <https://api.semanticscholar.org/CorpusID:59222730>.
- Matthew O’Kelly, Hongrui Zheng, Dhruv Karthik, and Rahul Mangharam. F1TENTH: An Open-source Evaluation Environment for Continuous Control and Reinforcement Learning. In *Proceedings of the NeurIPS 2019 Competition and Demonstration Track*, volume 123 of *Proceedings of Machine Learning Research*, pp. 77–89. PMLR, 08–14 Dec 2020.
- Hans B. Pacejka. Chapter 1 - tire characteristics and vehicle handling and stability. In Hans B. Pacejka (ed.), *Tire and Vehicle Dynamics (Third Edition)*, pp. 1–58. Butterworth-Heinemann, Oxford, third edition edition, 2012. ISBN 978-0-08-097016-5. DOI: <https://doi.org/10.1016/B978-0-08-097016-5.00001-2>. URL <https://www.sciencedirect.com/science/article/pii/B9780080970165000012>.
- Yunpeng Pan, Ching-An Cheng, Kamil Saigol, Keuntaek Lee, Xinyan Yan, Evangelos A Theodorou, and Byron Boots. Imitation learning for agile autonomous driving. *The International Journal of Robotics Research*, 39(2-3):286–302, 2020.
- Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 3803–3810. IEEE, 2018.

- 519 Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You Only Look Once: Unified,
520 Real-Time Object Detection. In *Proceedings of the IEEE Conference on Computer Vision and*
521 *Pattern Recognition (CVPR)*, June 2016.
- 522 Jonas Rothfuss, Bhavya Sukhija, Lenart Treven, Florian Dörfler, Stelian Coros, and Andreas Krause.
523 Bridging the sim-to-real gap with bayesian inference. In *2024 IEEE/RSJ International Conference*
524 *on Intelligent Robots and Systems (IROS)*, pp. 10784–10791, 2024. DOI: 10.1109/IROS58592.
525 2024.10801505.
- 526 Nikita Rudin, David Hoeller, Philipp Reist, and Marco Hutter. Learning to walk in minutes using
527 massively parallel deep reinforcement learning. In *5th Annual Conference on Robot Learning*,
528 2021. URL <https://openreview.net/forum?id=wK2fDDJ5VcF>.
- 529 Volkan Sezer and Metin Gokasan. A novel obstacle avoidance algorithm: "Follow the Gap Method".
530 *Robotics and Autonomous Systems*, 60(9):1123–1134, September 2012. ISSN 0921-8890. DOI:
531 10.1016/j.robot.2012.05.021.
- 532 Martin Simony, Stefan Milzy, Karl Amendey, and Horst-Michael Gross. Complex-YOLO: An Euler-
533 Region-Proposal for Real-time 3D Object Detection on Point Clouds. In *Proceedings of the*
534 *European Conference on Computer Vision (ECCV) Workshops*, September 2018.
- 535 Yunlong Song, Angel Romero, Matthias Müller, Vladlen Koltun, and Davide Scaramuzza. Reaching
536 the limit in autonomous racing: Optimal control versus reinforcement learning. *Science Robotics*,
537 8(82):eadg1462, 2023. DOI: 10.1126/scirobotics.adg1462. URL [https://www.science.](https://www.science.org/doi/abs/10.1126/scirobotics.adg1462)
538 [org/doi/abs/10.1126/scirobotics.adg1462](https://www.science.org/doi/abs/10.1126/scirobotics.adg1462).
- 539 Nitish Sontakke, Hosik Chae, Sangjoon Lee, Tianle Huang, Dennis W. Hong, and Sehoon Hal. Re-
540 sidual physics learning and system identification for sim-to-real transfer of policies on buoyancy
541 assisted legged robots. In *2023 IEEE/RSJ International Conference on Intelligent Robots and*
542 *Systems (IROS)*, pp. 392–399, 2023. DOI: 10.1109/IROS55552.2023.10342062.
- 543 Kyle Stachowicz, Dhruv Shah, Arjun Bhorkar, Ilya Kostrikov, and Sergey Levine. FastRLAP: A
544 System for Learning High-Speed Driving via Deep RL and Autonomous Practicing. In *Proceed-*
545 *ings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning*
546 *Research*, pp. 3100–3111. PMLR, 06–09 Nov 2023.
- 547 Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press,
548 second edition, 2018. URL [http://incompleteideas.net/book/the-book-2nd.](http://incompleteideas.net/book/the-book-2nd.html)
549 [html](http://incompleteideas.net/book/the-book-2nd.html).
- 550 Ben Tearle, Kim P. Wabersich, Andrea Carron, and Melanie N. Zeilinger. A Predictive Safety Filter
551 for Learning-Based Racing Control. *IEEE Robotics and Automation Letters*, 6(4):7635–7642,
552 2021. DOI: 10.1109/LRA.2021.3097073.
- 553 C. Thorpe, M.H. Hebert, T. Kanade, and S.A. Shafer. Vision and navigation for the Carnegie-Mellon
554 Navlab. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(3):362–373, 1988.
555 DOI: 10.1109/34.3900.
- 556 Sebastian Thrun, Mike Montemerlo, Hendrik Dahlkamp, David Stavens, Andrei Aron, James
557 Diebel, Philip Fong, John Gale, Morgan Halpenny, Gabriel Hoffmann, Kenny Lau, Celia
558 Oakley, Mark Palatucci, Vaughan Pratt, Pascal Stang, Sven Strohband, Cedric Dupont, Lars-Erik
559 Jendrossek, Christian Koelen, Charles Markey, Carlo Rummel, Joe van Niekerc, Eric Jensen,
560 Philippe Alessandrini, Gary Bradski, Bob Davies, Scott Ettinger, Adrian Kaehler, Ara Nefian,
561 and Pamela Mahoney. Stanley: The robot that won the DARPA Grand Challenge. *Journal of*
562 *Field Robotics*, 23(9):661–692, 2006. DOI: <https://doi.org/10.1002/rob.20147>.
- 563 Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Do-
564 main randomization for transferring deep neural networks from simulation to the real world. *2017*
565 *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 23–30, 2017.

- 566 Raphael Trumpp, Denis Hoornaert, and Marco Caccamo. Residual Policy Learning for Vehicle
567 Control of Autonomous Racing Cars. In *2023 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1–6,
568 2023. DOI: 10.1109/IV55152.2023.10186744.
- 569 Miguel Vasco, Takuma Seno, Kenta Kawamoto, Kaushik Subramanian, Peter R. Wurman, and Peter
570 Stone. A Super-human Vision-based Reinforcement Learning Agent for Autonomous Racing in
571 Gran Turismo. *Reinforcement Learning Journal*, 4:1674–1710, 2024.
- 572 Robin Verschueren, Gianluca Frison, Dimitris Kouzoupis, Jonathan Frey, Niels van Duijkeren, An-
573 drea Zanelli, Branimir Novoselnik, Thivaharan Albin, Rien Quirynen, and Moritz Diehl. acados
574 – a modular open-source framework for fast embedded optimal control. *Mathematical Program-
575 ming Computation*, 2021.
- 576 José L. Vázquez, Marius Brühlmeier, Alexander Liniger, Alisa Rupenyan, and John Lygeros.
577 Optimization-based hierarchical motion planning for autonomous racing. In *2020 IEEE/RSJ In-
578 ternational Conference on Intelligent Robots and Systems (IROS)*, pp. 2397–2403, 2020. DOI:
579 10.1109/IROS45743.2020.9341731.
- 580 Xin Wang, Yudong Chen, and Wenwu Zhu. A Survey on Curriculum Learning. *IEEE Transactions
581 on Pattern Analysis and Machine Intelligence*, 44(9):4555–4576, 2022. DOI: 10.1109/TPAMI.
582 2021.3069908.
- 583 Philipp Wu, Alejandro Escontrela, Danijar Hafner, Pieter Abbeel, and Ken Goldberg. DayDreamer:
584 World Models for Physical Robot Learning. In Karen Liu, Dana Kulic, and Jeff Ichnowski (eds.),
585 *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine
586 Learning Research*, pp. 2226–2240. PMLR, 14–18 Dec 2023.
- 587 Wenli Xiao, Haoru Xue, Tony Tao, Dvij Kalaria, John M. Dolan, and Guanya Shi. Anycar to
588 anywhere: Learning universal dynamics model for agile and adaptive mobility, 2024. URL
589 <https://arxiv.org/abs/2409.15783>.
- 590 Yufeng Yuan and A. Rupam Mahmood. Asynchronous Reinforcement Learning for Real-Time Con-
591 trol of Physical Robots. In *2022 International Conference on Robotics and Automation (ICRA)*,
592 pp. 5546–5552, 2022. DOI: 10.1109/ICRA46639.2022.9811771.
- 593 Yang Zhang, Zixiang Zhou, Philip David, Xiangyu Yue, Zerong Xi, Boqing Gong, and Hassan
594 Foroosh. PolarNet: An Improved Grid Representation for Online LiDAR Point Clouds Semantic
595 Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern
596 Recognition (CVPR)*, June 2020.
- 597 Jingyuan Zhao, Wenyi Zhao, Bo Deng, Zhenghong Wang, Feng Zhang, Wenxiang Zheng, Wanke
598 Cao, Jinrui Nan, Yubo Lian, and Andrew F. Burke. Autonomous driving system: A compre-
599 hensive survey. *Expert Systems with Applications*, 242:122836, 2024. ISSN 0957-4174. DOI:
600 <https://doi.org/10.1016/j.eswa.2023.122836>.
- 601 Wenshuai Zhao, Jorge Peña Queralta, and Tomi Westerlund. Sim-to-Real Transfer in Deep Rein-
602 forcement Learning for Robotics: a Survey. In *2020 IEEE Symposium Series on Computational
603 Intelligence (SSCI)*, pp. 737–744, 2020. DOI: 10.1109/SSCI47803.2020.9308468.
- 604 Qin Zou, Qin Sun, Long Chen, Bu Nie, and Qingquan Li. A Comparative Analysis of LiDAR
605 SLAM-Based Indoor Navigation for Autonomous Vehicles. *IEEE Transactions on Intelligent
606 Transportation Systems*, 23(7):6907–6921, 2022. DOI: 10.1109/TITS.2021.3063477.

Supplementary Materials

The following content was not necessarily subject to peer review.

A Extended Related Works

Autonomous driving research has evolved significantly, from early experiments in the 1970s to today’s sophisticated systems. Zhao et al. (2024) provide a comprehensive overview of these advancements, highlighting the field’s interdisciplinary nature. Foundational projects like the NavLab series (Thorpe et al., 1988) and the ARGO vehicle (Broggi et al., 1999) laid the groundwork for integrating technologies such as computer vision, machine learning, and advanced sensor systems into autonomous systems. These pioneering efforts were further accelerated by the DARPA Grand Challenges in 2004 and 2005, which catalyzed the adoption of machine learning and data-driven methods in autonomous vehicle development, driving the transition to computer-controlled vehicles with enhanced autonomy and intelligence (Thrun et al., 2006).

Various architectures have since been explored in autonomous driving. Layered architectures, with their modular design, excel in reliability and debugging, while end-to-end systems streamline processing through neural networks, though they face challenges related to data dependency and interpretability (Grigorescu et al., 2020). Significant progress has also been made in perception technologies. For example, Region-based Convolutional Neural Networks (Girshick et al., 2016) and single-stage methods like You Only Look Once (YOLO) and the Single-Shot Detector (SSD) have revolutionized image-based object detection (Redmon et al., 2016; Liu et al., 2016). Similarly, advancements in LiDAR-based systems, including Complex-YOLO and PointPillars, have enhanced 3D perception, critical for real-time navigation and obstacle avoidance (Simony et al., 2018; Lang et al., 2019; Zhang et al., 2020).

The evolution of SLAM systems underscores the increasing emphasis on accurate and efficient localization and mapping. Diverse projects demonstrate how vision-based techniques combined with inertial navigation improve robustness across diverse environments (Zou et al., 2022).

While autonomous driving technologies have matured for structured environments, recent research has shifted towards more dynamic domains like AR. This transition necessitates not only robust perception and planning but also advanced control algorithms capable of operating at the limits of vehicle dynamics. Classical control methods, such as PP and MPC, have served as reliable baselines. However, RL has emerged as a promising paradigm, offering superior adaptability and agility for racing tasks.

Reinforcement Learning in Simulation

In high-fidelity simulators like Gran Turismo, RL has demonstrated exceptional performance. For instance, Fuchs et al. (2021) and Vasco et al. (2024) outperformed human drivers using SAC and multi-step TD learning. While Fuchs et al. (2021) employed an observation space similar to ours, Vasco et al. (2024) incorporated a front-facing camera stream. By leveraging an asymmetric SAC architecture, their approach relied on proprioceptive observations during deployment, excluding information about upcoming track sections. Despite its success in simulation, this method has not yet demonstrated effective real-world transferability.

While RL methods often outperform classical approaches also in the FITENTH simulator, this performance fails to transfer to physical environments due to the sim-to-real gap, as highlighted in the works of Evans et al. (2023d). Factors such as model mismatch, sensor noise, and hardware delays exacerbate this gap, leading to instability and erratic behaviors in real-world deployments. Agents that outperformed a PP controller in simulation displayed rapid side-to-side swerving and instability during real-world tests, reaching lower speeds and ultimately lower lap times. These findings underscore the persistent challenges of sim-to-real transfer for RL-based autonomous racing systems.

655 **Trajectory-Aided Reinforcement Learning**

656 Trajectory-aided [RL](#) combines classical trajectory planning with [RL](#) to enhance safety, stability, and
657 performance in high-speed racing. Methods such as those proposed by [Ghignone et al. \(2024\)](#) and
658 [Evans et al. \(2023c\)](#) enforce adherence to precomputed trajectories by penalizing deviations. In
659 contrast, our approach primarily uses trajectory information to inform the [RL](#) agent about upcoming
660 track sections and define progress along the track, balancing flexibility and control in dynamic
661 environments.

662 **Model-Based Reinforcement Learning** Model-based [RL](#) has been always well regarded for its
663 sample efficiency, first demonstrated by [Deisenroth & Rasmussen \(2011\)](#) and more recently in the
664 [Model Based Policy Optimization \(MBPO\)](#) ([Janner et al., 2019](#)) and Dreamer ([Hafner et al., 2023](#))
665 architectures. Both these methods were shown to work well in real-world tasks, such as for drift-
666 ing maneuvers ([Rothfuss et al., 2024](#)), for locomotion and navigation ([Wu et al., 2023](#)), and on the
667 F1TENTH platform, where [Brunnbauer et al. \(2022\)](#) applied Dreamer to achieve faster lap times
668 in simulation compared to model-free methods. However, this last application exhibited bang-bang
669 control behavior when transferred to the real world, resulting in suboptimal performance. Addition-
670 ally, model-based methods require high computational demands and external workstations, mak-
671 ing this solution unsuitable for fully on-board learning on resource-constrained platforms like the
672 F1TENTH one, highlighting the need for more efficient on-car learning methods.

673 B Experimental Setup

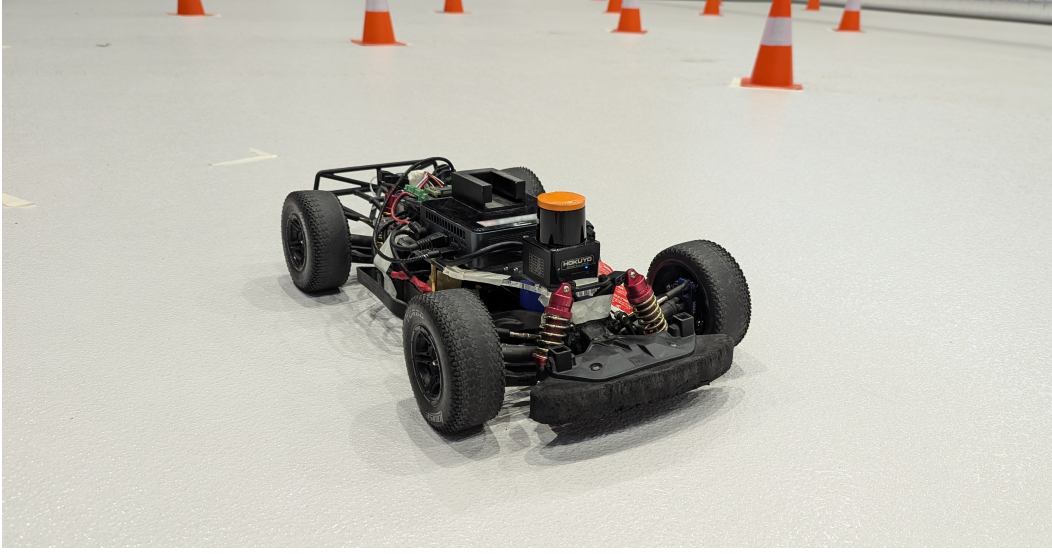


Figure 6: Image of the F1TENTH car used for evaluation equipped with the Intel NUC 10 computer (i7-10710U processor) and the Hokuyo UST-10LX [LiDAR](#) system.

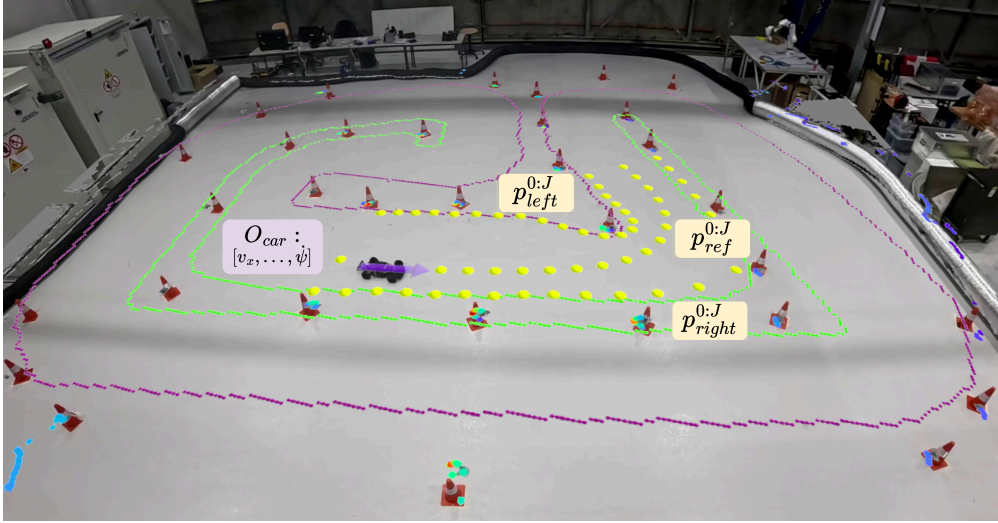


Figure 7: Image of C-track with an overlay of the car state observation $O_{\text{car}} = [v_x, v_y, \dot{\psi}, \Delta d, \Delta \psi, \delta_{\text{base}}, v_{\text{base}}, \delta_{\text{RL, prev}}, v_{\text{RL, prev}}]^T$, denoted as shortened state vector in purple. The observation of the track information $O_{\text{track}} = [\mathbf{p}_{\text{ref}}^{0:J}, \mathbf{p}_{\text{left}}^{0:J}, \mathbf{p}_{\text{right}}^{0:J}]^T$ is depicted in yellow. The virtual boundaries (green: inside, purple: outside), and [LiDAR](#) scans are further included. Track-boundary violations during evaluation are automatically detected based on the virtual boundaries. The cones are placed as reference points for the human viewers.

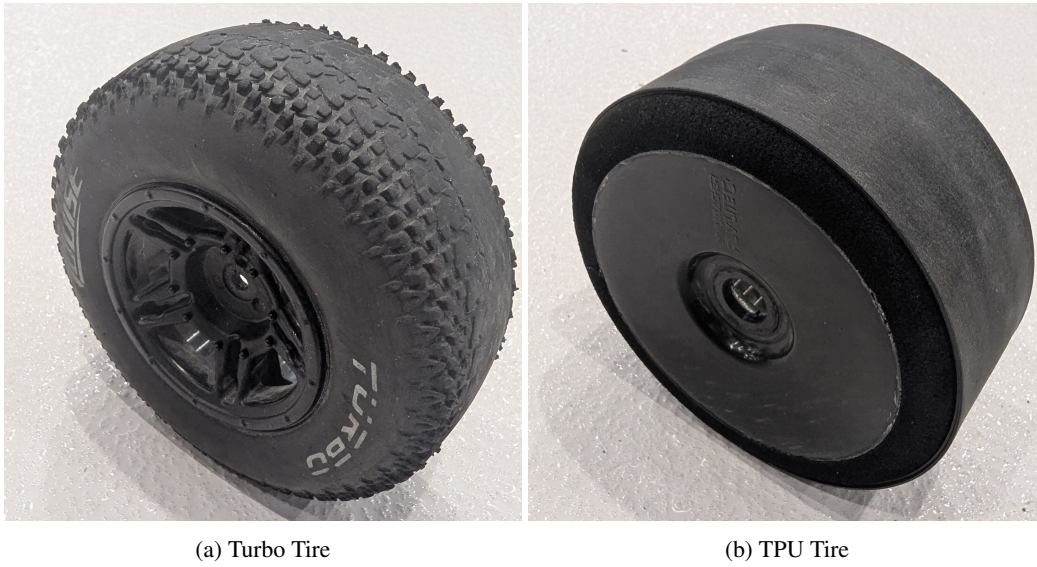


Figure 8: Images of the two tire types used during the experiments.

674 **C Controller Comparison**

Controller	t_{\min} [s] ↓	t_{\max} [s] ↓	t_{μ} [s] ↓	σ [s] ↓	n_{bound} ↓	T [min]
MAP Baseline	10.41	10.71	10.50	0.060	0	—
RL MAP	9.26	10.07	9.56	0.180	0	20.4
PP Baseline	12.06	12.18	12.13	0.033	0	—
RL PP	9.81	11.21	10.46	0.385	4	28.2
FTG Baseline	13.58	14.96	14.37	0.584	1	—
RL FTG	10.88	12.03	11.46	0.321	7	34.8
E2E RL	12.04	13.16	12.33	0.261	0	28.8
E2E RL	9.65	10.36	9.97	0.183	4	81.7

Table 2: Performance comparison of different baseline residual-controller combinations as well as the [E2E](#) controller on the `C-track` with Turbo tires after one battery life of training during 20 boundary-violation-free laps. t_{\min} represents the minimum lap time, t_{\max} the maximum lap time, t_{μ} the average lap time, σ the standard deviation of the lap times, n_{bound} the number of boundary violations until 20 violation-free laps are reached, and T the total training time.

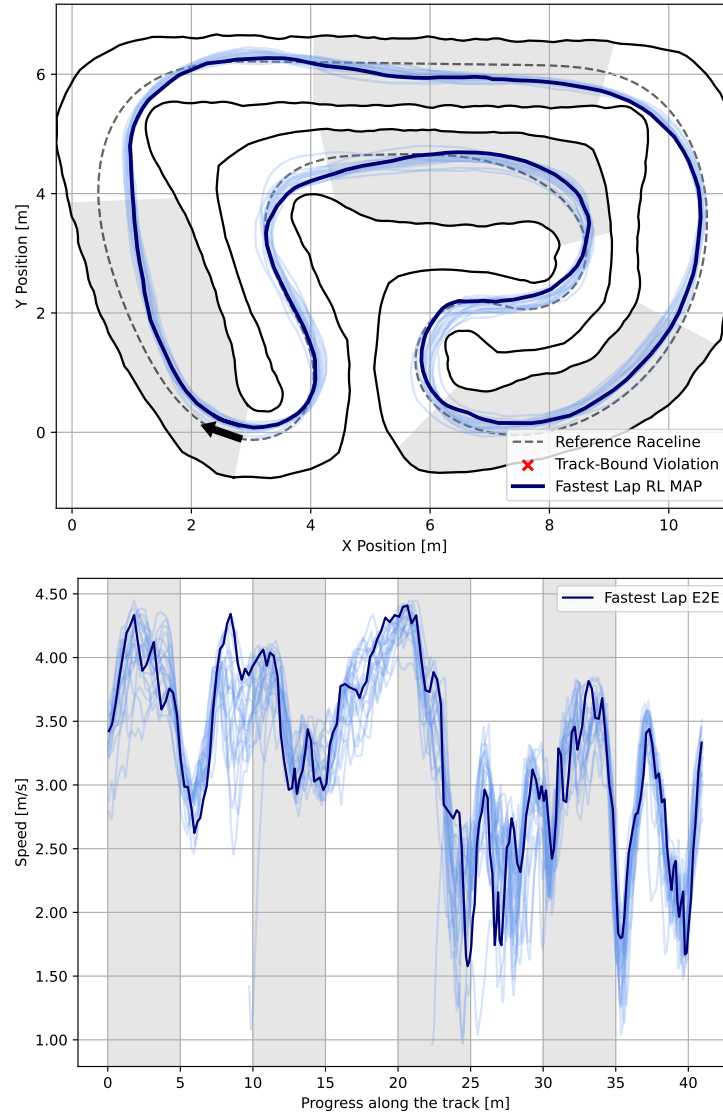
675 **D End-to-end Extended Results**

Figure 9: Visualizations of the trajectory and speed profile of the [E2E](#) agent after 29 min of training on the *C-track* with Turbo tires during deployment.

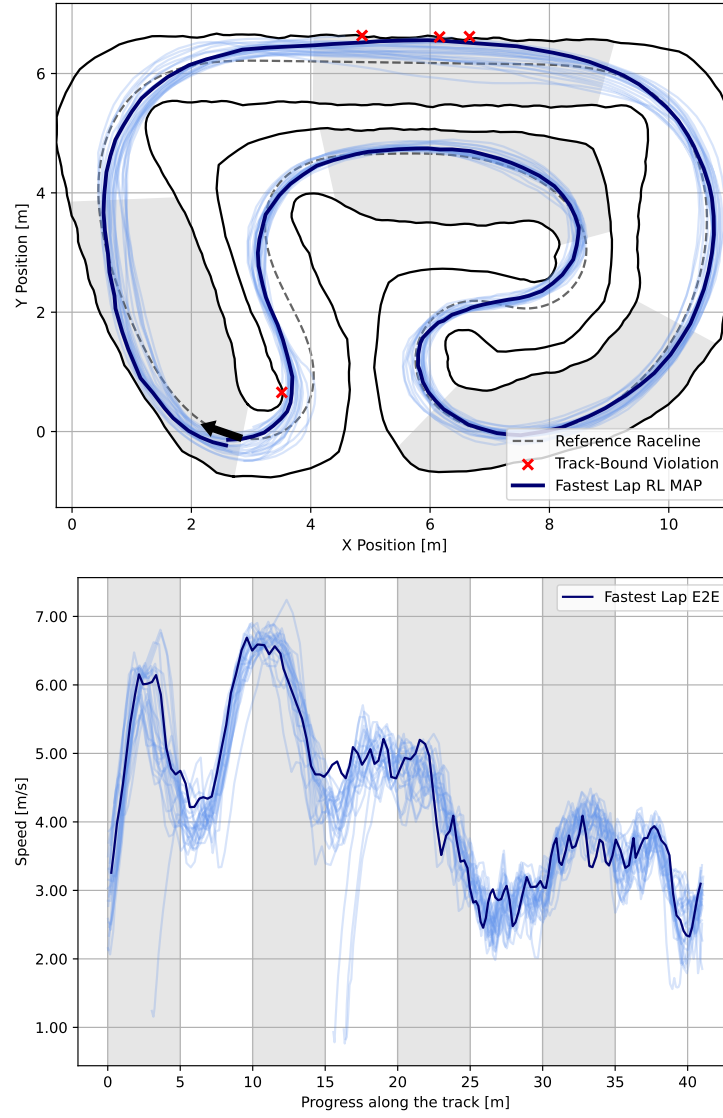


Figure 10: Visualizations of the trajectory and speed profile of the E2E agent after 82 min of training on the C-track with Turbo tires during deployment.

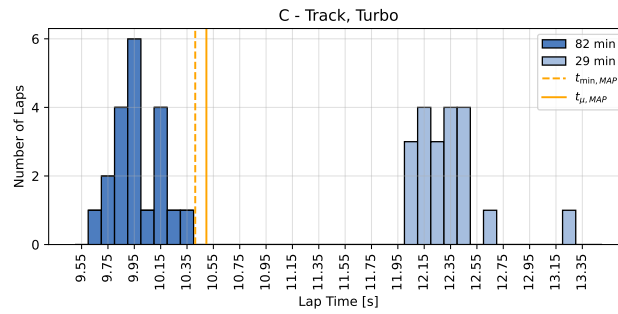


Figure 11: Histogram of lap times of the E2E controller after 29 min and 82 min. Orange lines represent the MAP baseline (Min. Lap Time: dashed, Avg. Lap Time: solid).

676 **E RL MAP Evaluation - Extended Results**

Run	t_{\min} [s] ↓	t_{\max} [s] ↓	t_{μ} [s] ↓	σ [s] ↓	n_{bound} ↓	T [min]
C-Track, Turbo tires						
MAP Baseline	10.41	10.71	10.50	0.060	0	—
Seed 1	9.46	10.07	9.67	0.184	0	22.1
Seed 2	9.26	9.89	9.54	0.183	0	20.4
Seed 3	9.28	9.67	9.47	0.111	2	20.4
Combined Depl.	9.26	10.07	9.56	0.181	2	21.0
C-Track, TPU tires						
MAP Baseline	12.93	13.78	13.45	0.338	0	—
Seed 1	12.26	12.59	12.42	0.084	0	32.7
Seed 2	12.36	12.77	12.55	0.133	0	33.4
Seed 3	12.23	12.96	12.54	0.215	1	33.2
Combined Depl.	12.23	12.96	12.50	0.163	1	33.1
Y-Track, Turbo Tires						
MAP Baseline	8.55	8.68	8.62	0.045	0	—
Seed 1	7.57	7.96	7.76	0.126	1	24.8
Seed 2	7.87	8.26	8.05	0.126	0	21.7
Seed 3	7.87	8.46	8.11	0.151	5	21.2
Combined Depl.	7.57	8.46	7.97	0.205	6	22.6

Table 3: Performance of **RL MAP** across different tire-track combinations and weight initializations. t_{\min} represents the minimum lap time, t_{\max} the maximum lap time, t_{μ} the average lap time, σ the standard deviation of the lap times, n_{bound} the number of boundary violations until 20 violation-free laps are reached, and T the total training time.

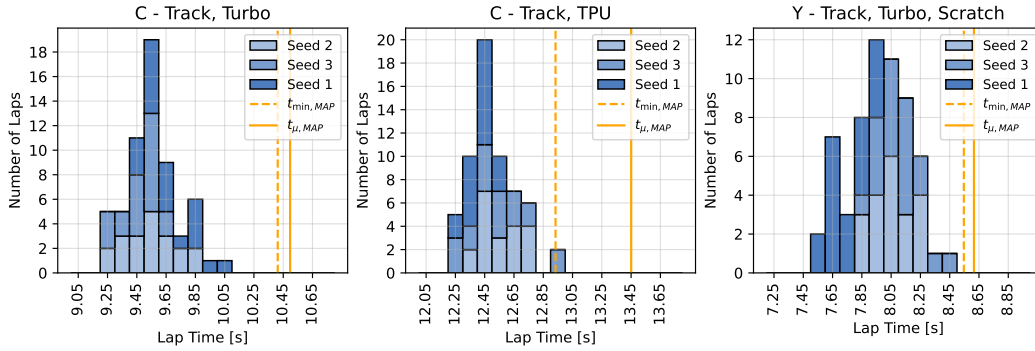


Figure 12: Stacked histogram of lap times across the different tire-track combinations and weight initializations, corresponding to Table 3. The orange lines represent the **MAP** baseline.

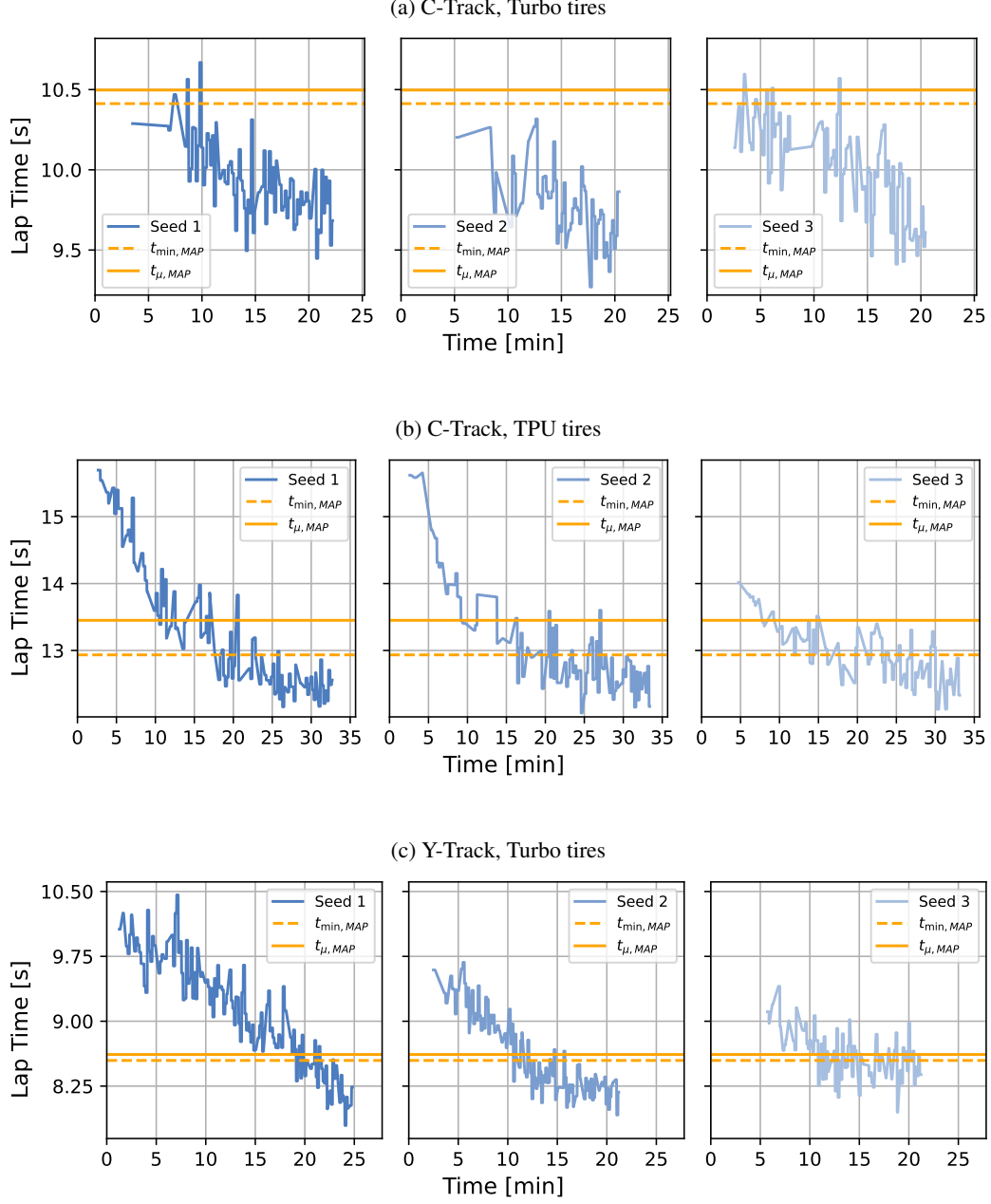


Figure 13: Lap Time vs. Wall Time during training for [RL MAP](#) across different track-tire combination and seed initialization, corresponding to table 3. The orange lines represent the [MAP](#) baseline.

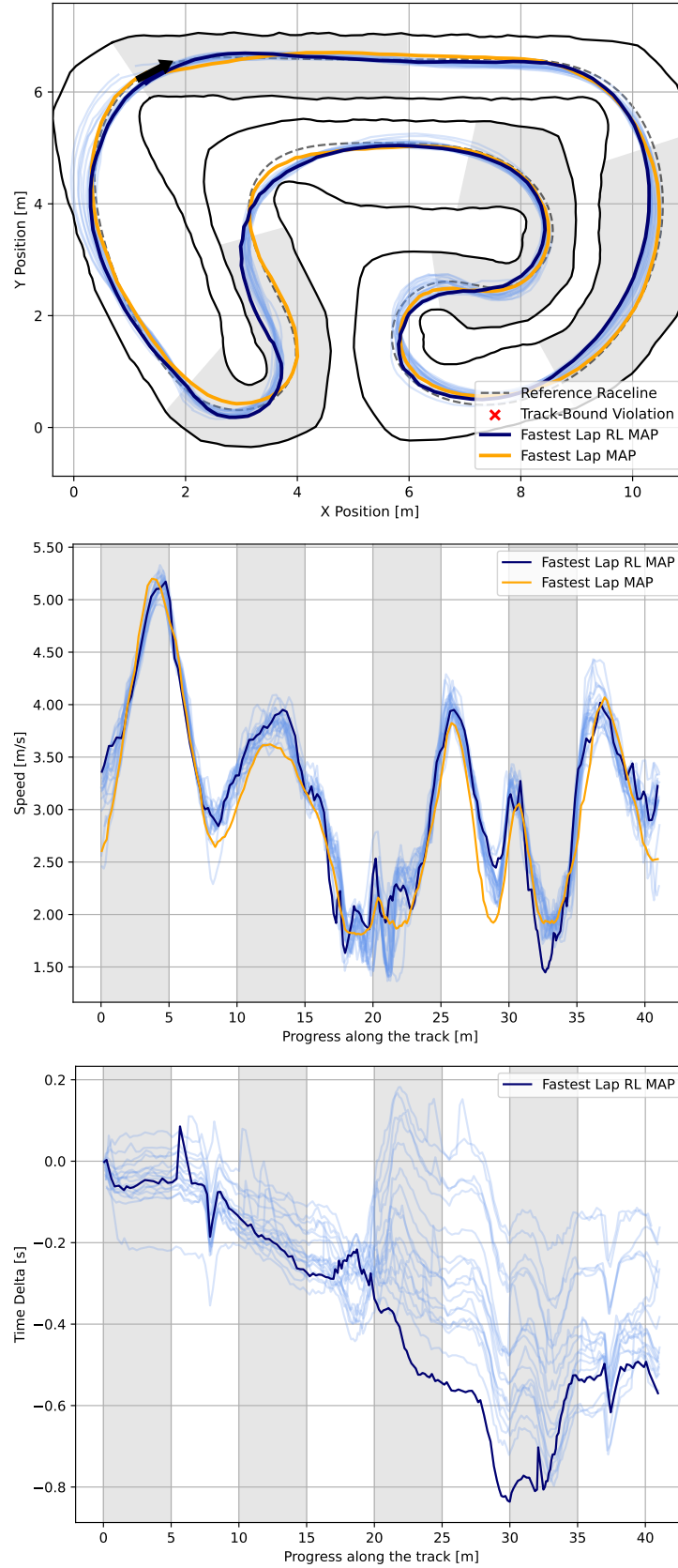


Figure 14: Visualizations of the trajectory, speed profile and time delta of **RL MAP** (Seed 2) after 20 minutes of training on the C-track with TPU tires.

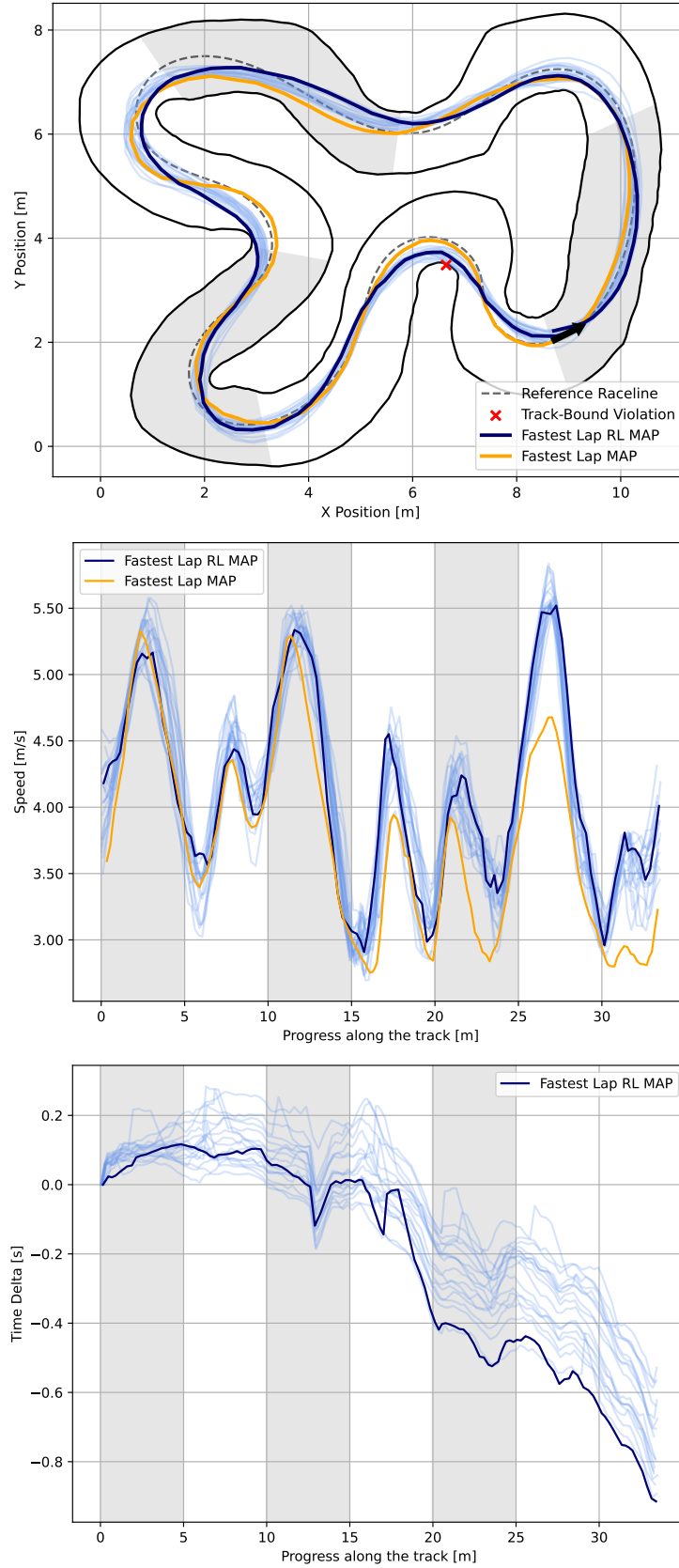


Figure 15: Visualizations of the trajectory, speed profile and time delta of **RL MAP** (Seed 1) after 20 minutes of training on the Y-track with Turbo tires.

677 F Generalization

678 Extended results, reporting every seed’s performance statistic on the generalization experiments are
 679 reported in Table 4.

680 **Learning from scratch** refers to the default training conducted in the rest of this work, and as such is
 681 a repetition of the results in Table 3, reported for ease of comparison. **Zero-Shot Transfer** refers to
 682 a policy trained on C-track and then deployed on Y-track without any additional training step,
 683 while **Few-Shot Transfer** refers to a policy trained on C-track and then deployed on Y-track
 684 after 20 minutes of additional training.

Run	t_{\min} [s] ↓	t_{\max} [s] ↓	t_{μ} [s] ↓	σ [s] ↓	n_{bound} ↓	T [min]
MAP Baseline	8.55	8.68	8.62	0.045	0	—
Learning from scratch						
Seed 1	7.57	7.96	7.76	0.126	1	24.8
Seed 2	7.87	8.26	8.05	0.126	0	21.6
Seed 3	7.87	8.46	8.11	0.151	5	21.2
Combined Depl.	7.57	8.46	7.97	0.205	6	22.6
Zero-Shot Transfer						
Seed 1	7.99	8.68	8.37	0.197	1	—
Seed 2	7.59	7.94	7.78	0.112	14	—
Seed 3	8.21	8.58	8.41	0.189	7	—
Combined Depl.	7.59	8.68	8.10	0.339	22	—
Few-Shot Transfer						
Seed 1	7.63	8.13	7.85	0.138	2	17.7
Seed 2	7.73	8.20	7.96	0.114	0	21.6
Seed 3	7.81	8.12	7.99	0.088	2	19.7
Combined Depl.	7.63	8.20	7.93	0.129	4	19.6

Table 4: Performance of **RL MAP** with few-shot and zero-shot transfer from the C-track to the Y-track. t_{\min} represents the minimum lap time, t_{\max} the maximum lap time, t_{μ} the average lap time, σ the standard deviation of the lap times, n_{bound} the number of boundary violations until 20 violation-free laps are reached, and T the total training time.

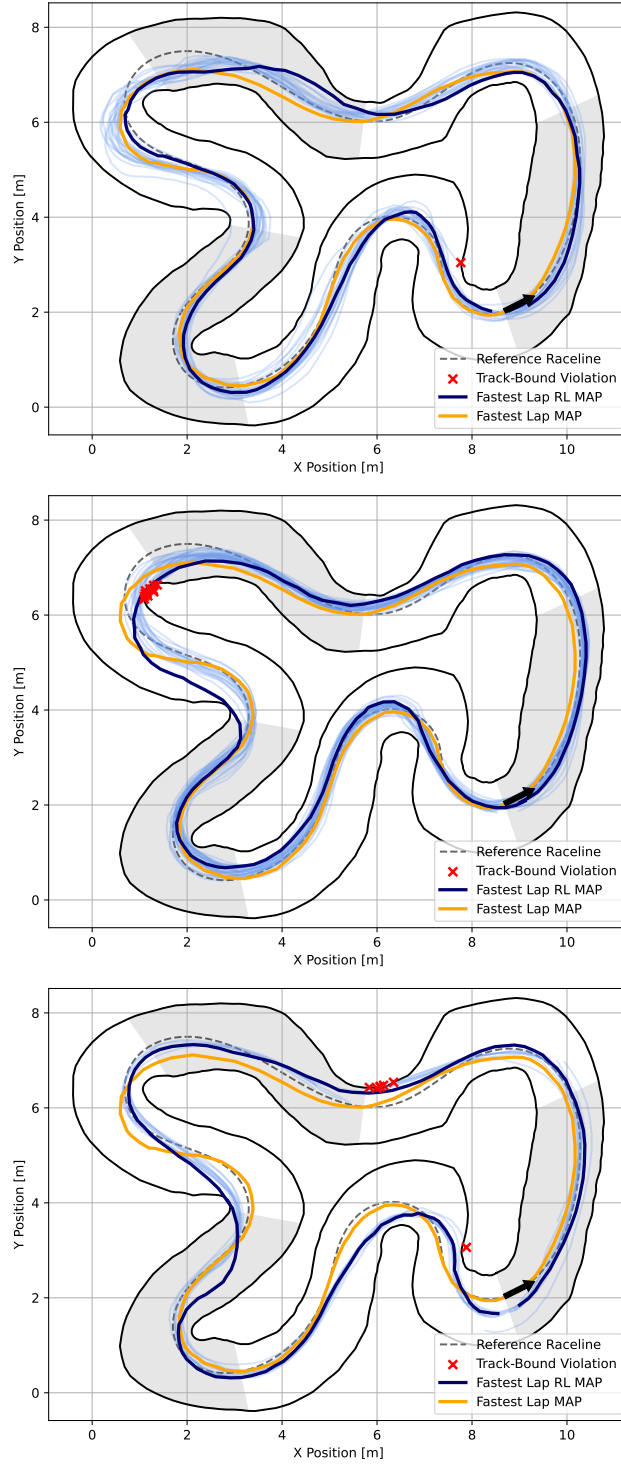


Figure 16: Visualizations of the trajectories during zero-shot transfer from C-track to Y-track for Seeds 1, 2 and 3 (from top to bottom).

685 **G Heuristic Delayed Reward Adjustment****Algorithm 1** Heuristic Delayed Reward Adjustment

Require: Replay buffer `replay_buffer`, current step index `current_step`

- 1: **Parameters:** `adjustment_steps` N , `penalty` p
- 2: **if** `terminal_state` **is** `True` **and** `replay_buffer.reward[current_step] = 0` **then**
- 3: **for** $n = 0$ **to** $N - 1$ **do**
- 4: `previous_step` \leftarrow `current_step` $- n$
- 5: `current_reward` \leftarrow `replay_buffer.reward[current_step]`
- 6: `decaying_penalty` $\leftarrow p - n(p/N)$
- 7: `adjusted_reward` \leftarrow `current_reward` $-$ `decaying_penalty`
- 8: `replay_buffer.reward[current_step - n]` \leftarrow `adjusted_reward`
- 9: **end for**
- 10: **end if**

686 **H Ablation Study Extended Results**

Run	t_{\min} [s] ↓	t_{\max} [s] ↓	t_{μ} [s] ↓	σ [s] ↓	$n_{\text{laps}, \mu}$	$n_{\text{bound}, \mu}$ ↓
HDRA+TD3, Sync	10.53	13.82	12.09	0.691	12.4	3.4
HDRA+TD3, Async	9.12	10.26	9.53	0.246	16.0	2.2
TD1, Async	9.73	20.81	11.53	2.823	11.0	5.6
TD3, Async	9.29	10.27	9.68	0.214	16.2	2.0

Table 5: Performance metrics summarized over five independent training runs, evaluating the last three minutes of 40 min-training runs during the ablation study on the simulated C-Track. The overall minimum, maximum, average lap time, and standard deviation are indicated with t_{\min} , t_{\max} , t_{μ} , σ respectively. The average over five runs of the number of laps and the number of boundary collisions in the three minutes is indicated with $n_{\text{laps}, \mu}$ and $n_{\text{bound}, \mu}$ respectively.

687 I Implementation Details

Category	Parameter	Value
Observations	Number of reference points J	20
	Look-ahead horizon l	6 m
Reward Design	Reward multiplier for progress λ	10
	Penalty p	10
	HDRA steps N	10
Safety Mechanisms	Safety filter ψ_{filter}	$[\pi/6; \pi/2]$ rad
	Increment/decrement rate ϵ	0.05 rad
SAC Parameters	Optimizer	Adam
	Learning rate	0.003
	Discount factor γ	0.96
	Temporal-difference (TD) steps	3
	Replay buffer size	$1e^6$
	Batch size	256
	Number of hidden layers	2
	Size of hidden layers	256
	Activations functions	ReLU

Table 6: Hyperparameters for on-board [RL](#).

688 J MPC Implementation Details

689 The [MPC](#) formulation used in this work uses a single-track bicycle model, with tire forces modeled
690 with the *Pacejka Magic Formula* ([Pacejka, 2012](#)). The cost function is derived from the [MPC](#) for-
691 mulation of [Vázquez et al. \(2020\)](#).

692 The model, expressed in curvilinear coordinates, reads as follows:

$$\dot{s} = ((v_x \cos \varphi_{ref}) - (v_y \sin(\varphi_{ref}))) / (1 - \kappa_{ref}(s) n) \quad (2)$$

$$\dot{n} = v_x \sin \varphi_{ref} + v_y \cos \varphi_{ref} \quad (3)$$

$$\dot{\varphi}_{ref} = r - \kappa_{ref}(s) \dot{s} \quad (4)$$

$$\dot{v}_x = a + \frac{1}{m}(-F_{yf} \sin \delta + m v_y r) \quad (5)$$

$$\dot{v}_y = \frac{1}{m}(F_{yr} + F_{yf} \cos \delta - m v_x r) \quad (6)$$

$$\dot{r} = \frac{1}{I_z}(F_{yf} l_f \cos \delta - F_{yr} l_r) \quad (7)$$

693 where s represents the longitudinal advancement along the reference line, n the perpendicular lateral
694 displacement, φ_{ref} the angle relative to the tangent at the point that meets the perpendicular, and
695 $\kappa_{ref}(s)$ the curvature of the reference line at the corresponding s . The longitudinal input is modeled
696 as a and the steering is δ , resulting in the input $\mathbf{u} = (a, \delta)$. Longitudinal and lateral velocities
697 are represented by v_x , v_y , respectively, and r represents the yaw rate. These latter three states are
698 mostly influenced by tire forces, F_{yf} , F_{yr} , indicating the lateral tire force on the front and rear
699 lumped tires, respectively. They follow the equations from the *Pacejka Magic Formula*:

$$F_{y,i} = \mu F_{z,i} D_i \sin[C_i \arctan(B_i \alpha_i - E_i (B_i \alpha_i - \arctan(B_i \alpha_i)))] \quad (8)$$

where $i \in \{front, rear\}$ discriminates between front and rear tire. The slip angles α_i are as follows:

$$\alpha_f = \arctan\left(\frac{v_y + rl_f}{v_x}\right) - \delta \quad (9)$$

$$\alpha_r = \arctan\left(\frac{v_y - rl_r}{v_x}\right). \quad (10)$$

The *Pacejka* parameters (B_i, C_i, D_i, E_i) are obtained via the procedure detailed by [Dikici et al. \(2025\)](#).

The *MPC* cost function, derived from [Vázquez et al. \(2020\)](#), is designed in order to follow the raceline (both on the positional and the velocity profile) computed by adjusting the minimum curvature raceline presented by [Heilmeier et al. \(2020\)](#) to the FITENTH vehicle. The *MPC* problem then approximately reads as follows:

$$\min_U \quad l_f(\mathbf{x}_N) + \sum_{i=0}^{N-1} l(\mathbf{x}_i, \mathbf{u}_i) \quad (11)$$

$$\mathbf{x}_{i+1} = f(\mathbf{x}_i, \mathbf{u}_i) \quad \forall i \in \{0, \dots, N-1\}$$

$$\mathbf{x}_i \in \mathcal{X} \quad (12)$$

$$\mathbf{u}_i \in \mathcal{U} \quad (13)$$

$$\mathbf{x}_N \in \mathcal{X}_f \quad (14)$$

$$\mathbf{x}_0 = \hat{\mathbf{x}} \quad (15)$$

where:

- The state is $\mathbf{x} = [s, n, \varphi_{ref}, v_x, v_y, r]^\top$
- The number of timesteps is represented by N
- The state transition equation, derived from a *Runge-Kutta* discretization of the equations above, is displayed as $f(\mathbf{x}, \mathbf{u})$
- \mathcal{X} represents the track boundaries and state constraints sets
- \mathcal{U} represents the input constraints set
- $\hat{\mathbf{x}}$ is the measured state
- the cost function is $l(\mathbf{x}_i, \mathbf{u}_i) = w_n n + w_{\varphi_{ref}} \varphi_{ref} + w_{v_x} (v_x - v_x^{race})$, with v_x^{race} being the velocity obtained from the raceline generation
- $l_f(\mathbf{x}_i, \mathbf{u}_i) = 10l(\mathbf{x}_i, \mathbf{u}_i)$

Soft constraints are further integrated on state constraints, to try to ensure feasibility. The controller then is manually tuned for lap time minimization. The implementation is then carried out within the *acados* framework ([Verschuere et al., 2021](#)).

The *MPC* controller, in the end, did not achieve the fastest performance on the tires with the highest grip, and only achieved the fastest average lap time on the tires with the lowest grip. We attribute this performance gap to potential *sim-to-real* mismatches, such as unmodelled delays, as well as the higher signal noise inherent to the vehicle's sensor setup and embedded system constraints, which may have compromised the *MPC*'s performance and induced unnecessary conservativity.