AI Open 7 (2026) 123–132

Contents lists available at ScienceDirect

AI Open

journal homepage: www.elsevier.com/locate/aiopen

Full length article

Human professional level driving agent for race car simulation environments

Gergely Bári [a,*], László Palkovics [b,c]

[a] HUMDA Lab nonprofit Kft., Dávid Ferenc u. 4-6., Budapest, H-1113, Hungary
[b] Systems and Control Laboratory, Institute for Computer Science and Control (SZTAKI), Kende utca 13-17., Budapest, H-1111, Hungary
[c] Pekár Imre Doctoral School of Mechanical Engineering, University of Debrecen, Böszörményi út 138., Debrecen, H-4032, Hungary

ARTICLE INFO

Keywords:
Deep learning
Reinforcement learning
Modeling
Simulation
Vehicle dynamics
Autonomous driving
Race car driving

ABSTRACT

Precise vehicle control at the limits of tire adhesion is paramount for both competitive motorsport performance and the safe execution of emergency maneuvers in road vehicles. Mastering this "grip-limit driving" presents significant challenges due to highly non-linear vehicle dynamics and sensitivity to changing conditions, often exceeding the capabilities of traditional controllers and driver models. This paper investigates the efficacy of Deep Reinforcement Learning (DRL), specifically the Proximal Policy Optimisation (PPO) algorithm, as a data-driven approach to learn expert-level driving skills within the TORCS high-fidelity race car simulation environment. An agent was trained end-to-end, utilizing "realworld-friendly" state signals (such as speeds, accelerations, and yaw rate, simple LiDaR, etc.) as input to determine continuous steering and pedal commands. Notably, the trained Agent achieved lap times comparable to a human e-sport world champion on the target track, demonstrating the potential of this methodology while also highlighting how agents can exploit idealized simulation to achieve superhuman control. Furthermore, this work presents the formulation of the time-optimal driving task as a DRL problem and offers a novel justification for the commonly used "progress reward" function, demonstrating its conceptual link to the time-difference feedback mechanisms human drivers use for performance optimization. These findings provide valuable insights into AI-driven vehicle control under extreme conditions and contribute to the development of more capable autonomous agents for simulation and potentially, real-world applications.

1. Introduction

The ability to precisely control a vehicle under demanding conditions, particularly near the limits of tire adhesion, represents a critical capability in automotive engineering. In motorsport, operating at this "grip limit" is fundamental to achieving competitive lap times. For everyday road vehicles, mastering control during extreme transients is essential for navigating emergency situations and enhancing active safety systems. However, accurately predicting and controlling vehicle behavior near the friction boundary is inherently challenging. It involves complex interactions between tires and the road surface, governed by highly non-linear dynamics that are sensitive to variations in speed, load, and environmental factors, making analytical modeling and control design non-trivial tasks.

Traditional control strategies, such as Model Predictive Controller (MPC), have shown success in autonomous racing but often rely heavily on the accuracy of underlying vehicle models, which can be difficult to guarantee across the full range of limit-handling scenarios. Furthermore, within vehicle development, simulation tools like Driver in the Loop (DiL) simulators are increasingly vital. Yet, the classical driver models often employed within these simulations frequently lack the fidelity to replicate the adaptive, nuanced control inputs of expert human drivers operating at the vehicle's limits, thus limiting their value. Concurrently, while Reinforcement Learning (RL) offers a promising data-driven paradigm, its direct application to safety-critical, real-world driving tasks faces substantial hurdles related to sample efficiency, safety during exploration, and sim-to-real transfer.

This research explores Deep Reinforcement Learning (DRL) as a powerful methodology capable of overcoming some of these limitations, particularly within the controlled environment of simulation. DRL agents can learn sophisticated control policies directly through trial-and-error interaction, implicitly capturing complex system dynamics without requiring an explicit, perfectly accurate model. Specifically, this paper focuses on employing the Proximal Policy Optimisation (PPO) algorithm, a robust and widely-used DRL technique, within the open-source TORCS racing simulator. We train an end-to-end agent whose policy network receives readily available vehicle dynamic state

* Corresponding author.
E-mail address: gergely.bari@fun-ke.com (G. Bári).
[1] https://youtu.be/_0XoIlewhh4.

https://doi.org/10.1016/j.aiopen.2026.02.007
Received 21 April 2025; Received in revised form 24 February 2026; Accepted 27 February 2026
Available online 28 February 2026

information (e.g., wheel speeds, longitudinal and lateral velocities and accelerations, yaw rate, track position relative angle) and outputs continuous commands for steering wheel angle and throttle/brake position, aiming to minimize lap time on a given racetrack. In this paper, we define "human-professional-level" driving not merely by a single performance metric, but by a combination of two key criteria. Firstly, the agent must achieve final lap times that are quantitatively comparable to those of a verified human e-sport champion under almost identical conditions. Secondly, and more critically, the agent must demonstrate the *emergent learning* of complex control strategies characteristic of expert drivers, without these behaviors being explicitly encoded. This includes techniques such as effective threshold braking to maximize deceleration without locking the wheels, utilizing the full width of the track to optimize the racing line (the "out-in-out" principle), and executing rapid transitions between full throttle and full brake. The successful learning of these nuanced behaviors from a simple, high-level reward signal is a primary focus of our investigation. This paper makes several contributions to the field. Firstly, we demonstrate that the proposed DRL agent, trained using PPO and vehicle state observations, can successfully learn to operate at the grip limit and achieve lap times comparable to a human e-sport world champion. This result validates the effectiveness of the approach while simultaneously revealing how the agent leverages idealized actuator models within the simulation to attain its performance. Secondly, we provide a detailed comparative analysis contrasting the emergent driving style of the AI agent with that of the human champion. While achieving similar performance levels, the analysis reveals distinct differences in control strategies, input characteristics (e.g., steering smoothness and rate), and how each handles the specific dynamics of the simulated vehicle. Thirdly, we offer a novel theoretical justification for the "progress-reward" function, commonly used in DRL-based racing agents, by showing its direct connection how human drivers learn and optimize their track performance. A video comparison of laps driven by the human champion (after ∼8 h of practice) and the AI agent is available for viewing online, providing a direct visual illustration of their distinct driving styles.[1]

Our work also serves to provide complementary evidence to recent successes in DRL-based racing (Fuchs et al., 2021; Wurman et al., 2022), investigating whether similar performance can be achieved in a different simulation environment and with a distinct, vehicle-state-based input modality.

The findings presented herein hold significance for advancing the development of high-fidelity AI drivers for automotive simulation and testing platforms. The insights gained from comparing AI and human driving strategies can inform the design of more robust and human-aware Advanced Driver Assistance Systems (ADAS) and emergency control systems for road vehicles. This paper is structured as follows: Section 2 reviews relevant prior work. Section 3 details the DRL problem formulation, the agent architecture, the simulation environment, and the derivation of the reward function. Section 4 presents the simulation results, including the learning process and the in-depth comparison between the agent and the human driver. Finally, Section 5 concludes the paper, summarizing the key findings, acknowledging limitations, and discussing promising avenues for future research.

2. Literature study

The challenge of controlling a racecar at its handling limits is addressed in the scientific literature from multiple perspectives, sometimes framed as "race car driver modeling" and other times as "vehicle control on the grip limit". Fundamentally, both aim to replicate or achieve expert-level driving performance near the boundaries of tire adhesion. While classical control techniques have long been applied, the advent of Deep Reinforcement Learning (DRL) represents a more recent, data-driven paradigm shift in tackling this complex problem. Most early studies focused on developing subsystems within the traditional

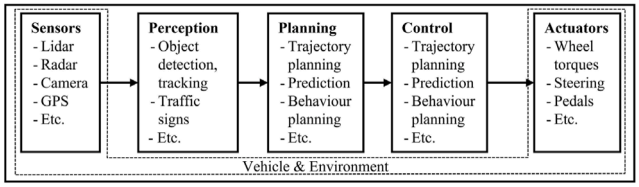

Fig. 1. Widely used decomposition of the self driving problem (Bári, 2018).

autonomous driving decomposition (see Fig. 1), such as trajectory planning or feedback control loops. Learning-based, particularly end-to-end, approaches have gained significant traction more recently, leveraging the ability of ML to handle complex, non-linear dynamics implicitly. In the following, we first summarize relevant classical control approaches before delving into the evolving landscape of RL methods.

Among the most advanced non-learning-based methods, Model Predictive Controller (MPC) and Robust Control techniques are prevalent. Verschueren et al. (2014) and Cataffo et al. (2022) show that MPC excel in racing by optimizing performance within predicted constraints, while robust control methods like H_∞ are often preferred for road vehicles where managing uncertainty and guaranteeing stability across varied conditions is paramount. Pedone and Fagiolini (2023) present a discrete-time observer and controller focused on disturbance rejection for lateral control, prioritizing robustness over pure lap time optimization. Similarly, Zhu et al. (2024) emphasize model-based strategies with cascaded control for transient dynamics management in high-speed scenarios such as the Indy Autonomous Challenge. Brown and Gerdes (2020) apply nonlinear MPC to coordinate tire forces during aggressive maneuvers, while Cataffo et al. (2022) use MPC for lap time minimization in small-scale racers. However, these classical methods often rely on accurate vehicle models, which are hard to obtain for the full non-linear range of grip-limit driving, and they may struggle to replicate the intuitive risk management and adaptive skills of human experts, particularly for realistic driver modeling in simulations like DiL.

RL methods offer an alternative by learning control policies directly from interaction. Prior work varies considerably in (i) the input observations (e.g., vision-based approaches (Bári and Palkovics, 2026; Vasco et al., 2024; Venkatesh et al., 2021; Nashashibi, 2018), versus vehicle-dynamics signals as used in this work), (ii) the learning algorithm (e.g., model-free methods such as SAC (Haarnoja et al., 2018), PPO (Schulman et al., 2017), and DDPG (Silver et al., 2014), versus model-based methods such as PlaNet (Hafner et al., 2019) and Dreamer (Hafner et al., 2025; Brunnbauer et al., 2022)), and (iii) the simulation environment (e.g., high-fidelity simulators such as Gran Turismo (Fuchs et al., 2021; Wurman et al., 2022; Lee et al., 2025), Assetto Corsa (Remonda et al., 2024), and VSM (Remonda et al., 2021); open-source options such as TORCS (Remonda et al., 2022), which is also used here; and simpler platforms such as F1TenthGym (Brunnbauer et al., 2022).

For example, Remonda et al. (2022) employed DDPG within the TORCS environment using vehicle telemetry but focused on generalization to unknown tracks rather than achieving verified human-champion performance levels. In a follow-up work Remonda et al. (2021) used a professional simulator (VSM) and DDPG, demonstrating that an RL agent could outperform human drivers; however, this study potentially relied on inputs difficult to obtain outside simulation (e.g., tire saturation) and the benchmark human performance level was not explicitly defined as world champion caliber. In contrast, the present approach utilizes signals potentially easier to obtain in real-world scenarios and compares directly against verified e-sport world champion data.

Vision-based approaches have also shown significant success. Lee et al. (2025) achieved superhuman performance in Gran Turismo 7 using only camera images and vehicle data during execution, employing

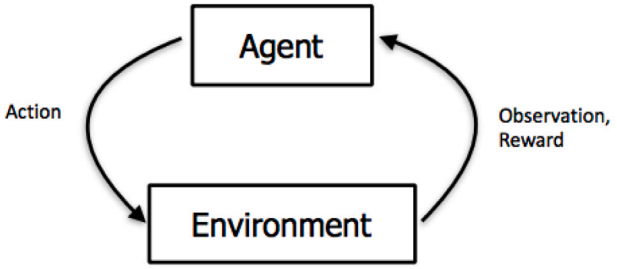

Fig. 2. Scheme of the Agent - Environment interaction in Reinforcement Learning problems.

an asymmetric actor-critic architecture. This contrasts with our work's exploration of using vehicle dynamic signals directly as the primary input modality. Earlier work in Gran Turismo by Fuchs et al. (2021) (single-car time trials using SAC) and Wurman et al. (2022) (head-to-head racing) also demonstrated agents surpassing human capabilities. While the approach of Fuchs et al. (2021) is conceptually close to ours in its goal, our research provides a direct test of the generality of such findings by replicating the high-level performance in a different simulation environment (TORCS vs. Gran Turismo) with a different model-free algorithm (PPO vs. SAC). Crucially, we also provide a novel derivation connecting the reward structure used in works like Fuchs et al. (2021) to human learning principles.

Other relevant research includes comparisons between model-based (e.g., Dreamer) and model-free RL for scaled racing cars, where model-based methods showed advantages in sample efficiency and generalization (Brunnbauer et al., 2022) Our work provides a counterpoint focused on a model-free approach in full-scale simulation. Venkatesh et al. (2021) presented a hybrid vision-based method combining unsupervised trajectory generation and supervised learning, differing from our pure RL optimization approach. Building upon earlier vision-based RL efforts like the work of Nashashibi (2018) (which used A3C in a rally game), our work utilizes a different algorithm (PPO) and focuses specifically on vehicle dynamic state inputs within the TORCS environment. Finally, the development of benchmark platforms like the one based on Assetto Corsa in the work of Remonda et al. (2024) represents a valuable resource for testing and comparing different autonomous racing algorithms, including approaches similar to ours, across various high-fidelity simulation contexts.

3. Methods

For the goal of this work, the task of driving a race car is transformed into a Reinforcement Learning (RL) problem following the general RL approach pictured in Fig. 2. The resulting RL problem then was implemented using the Stable Baseline3 package (Raffin et al., 2021).

In the scope of this work the **Agent** is a single Neural Network (NN). This NN has a special type, it is called Multilayer Perceptron (MLP). The MLP used in this work was implemented in the StableBaselines3 package. It has three layers that are common for both the so-called value network and policy network (with 300, 600, 600 neurons respectively), and one additional layer (with 600 neurons) for the value network only, all with ReLU activations.

The **Action** decided by this agent consists of two real numbers in the range of [−1..1]. One of these represents the scale of maximum throttle (+1) to maximum brake (−1), while the other represents the steering wheel angle from maximum left (+1) to maximum right (−1). It is typical to restrict the control abilities of the agent to the level of a human, which usually involves the implementation of a rate limit on the actions; however, this was not applied in this work. Here, the agent-environment interaction occurred at a rate of 20 Hz, while the control frequency of a professional race car driver is estimated to be around 10 Hz. This discrepancy was accepted and taken into account when interpreting the results.

The **Environment** used in this work is the open-source car racing simulation software called TORCS (OpenSource, 2020). TORCS provides a straightforward and adaptable structure for testing AI techniques. As a detailed car simulation, TORCS can provide vehicle dynamic signals and visual 3D representations. In this work, a set of vehicle dynamic states are used as the **Observation** (see Table A.4 in Appendix for details), while another set of these signals were used to form the appropriate **Reward** for training the RL agent.

This work uses a single NN as the **Agent**, applying a so-called "end-to-end" method. Although the choice of distinct Perception-Planning-Control subsystems seems advantageous for road cars, it can pose challenges in racing. Planning *safe trajectories* for road cars typically assumes worst-case grip values. However, planning a *time optimal trajectory* in racing necessitates a precise (<1%) grip estimate. Achieving such precision is especially difficult in transient conditions, where grip estimation techniques are often only accurate to around ∼10% (Acosta and Kanarachos, 2018). This high sensitivity to grip level (a 1% change can alter lap times by 0.15% (Kelly and Sharp, 2010)) means race drivers perform not just optimal control but also crucial risk management based on intuition. As the landmark achievements in the work of Silver et al. (2016) suggest, agents can develop analogous intuition, hence modeling this risk management via deep learning presents a viable alternative to traditional approaches.

In Reinforcement Learning, the **Reward** is a numerical signal the agent aims to maximize over time via its actions. Constructing an effective reward function is crucial; its density (frequency of feedback) significantly impacts learning stability and speed. Using only the final lap time provides a very sparse reward, leading to unstable and slow learning, as feedback arrives only after potentially thousands of steps. To create a dense reward (available at each step), much relevant literature (Fuchs et al., 2021; Wurman et al., 2022) utilizes a "progress reward" – the distance traveled along the track centerline in the most recent timestep. We adopt this form, as shown in Eq. (1):

$$r^{progress} = s_t^{cl} - s_{t-1}^{cl} \tag{1}$$

where s_t^{cl} is the distance traveled along the track centerline until timestep t.

Although this reward structure is effective, we find it valuable to demonstrate its connection to the learning process of human race car drivers, a justification seemingly absent in prior literature. For human drivers, a primary feedback signal during practice is the real-time time-difference displayed on their dashboard. This signal compares the current lap's progress against a reference lap (typically their fastest), indicating the time gained or lost upon reaching a specific track position (s_t^{cl}). Drivers use the change in this time-difference (dt_t) to evaluate different strategies (e.g., cornering lines). Basing the reward on the change in this time advantage, appears intuitive.

To start constructing this reward, consider a racecar moving along some racing line on a race track. Let s_t^{cl} be the distance traveled by this racecar along the centerline of the track at time step t. Consider also an other, so-called "reference car" that is moving along the centerline of the same racetrack with a constant v^{ref} speed. The time to travel the distance s_t^{cl} will take t for the racecar, while it will take t_t^{ref} for the reference car, where

$$t_t^{ref} = \frac{s_t^{cl}}{v^{ref}} \tag{2}$$

Let dt_t be the time advantage of the agent reaching the position s_t^{cl}, compared to the reference car.

$$dt_t = t_t^{ref} - t \tag{3}$$

The proposed reward $r_t^{t_{diff}}$ should award (>0) the behavior that results in dt_t increasing, which means dt_t should be greater than dt_{t-1}

$$r_t^{t_{diff}} = dt_t - dt_{t-1} \tag{4}$$

Substituting (3) into (4) yields:

$$r_t^{t_{diff}} = t_t^{ref} - t - \left(t_{t-1}^{ref} - t_{t-1}\right) \tag{5}$$

Using (2) in (5) gives:

$$r_t^{t_{diff}} = \frac{s_t^{cl}}{v^{ref}} - t - \left(\frac{s_{t-1}^{cl}}{v^{ref}} - t_{t-1}\right) \tag{6}$$

After some arrangements in (6) yields:

$$r_t^{t_{diff}} = \frac{s_t^{cl} - s_{t-1}^{cl}}{v^{ref}} - ts \tag{7}$$

Here $ts = t - t_{t-1}$ is the timestep. As the speed of the reference car v_{ref} is constant and free of choice, and the timestep ts is constant and independent of agent actions, (7) can be reduced to:

$$r_t^{t_{diff}} = s_t^{cl} - s_{t-1}^{cl} \tag{8}$$

That is exactly the same as $r^{progress}$ in (1). To enhance training robustness, additional reward components were incorporated. A large positive reward was given upon successful lap completion. Penalties (r_t^{ter}) were applied for detrimental events such as leaving the track, turning backward, damaging the car, moving backward along the track centerline, or making insufficient progress within a defined time window (specific penalty values are listed in Table A.2). Furthermore, an action penalty term (r_t^{act}) was added to discourage excessively large or non-feasible action outputs from the neural network, improving training stability:

$$r_t^{act} = \left(\frac{|a_t|}{p^{sc}} - p^{bnd} + 1\right)^2 \tag{9}$$

Here, a_t represents the action vector at time t, and p^{sc} and p^{bnd} are scaling and boundary parameters, respectively (see Table A.2). The final reward function used for training is the sum of these components:

$$r_t = r_t^{t_{diff}} + r_t^{ter} - r_t^{act} \tag{10}$$

Each component of this composite reward function plays a distinct role in shaping the agent's behavior. The progress reward, $r_t^{t_{diff}}$, is the primary driver for performance, directly incentivizing the agent to maximize its speed along the track centerline. The termination penalties, r_t^{ter}, are sparse but high-magnitude signals that act as hard constraints, efficiently pruning behaviors that are unsafe (eg.: leaving the track) or counter-productive (eg.: going backward). Finally, the action penalty, r_t^{act}, serves as a regularization term that promotes control inputs within the valid actuator range (±1).

3.1. PPO algorithm and loss function

The agent was trained using the Proximal Policy Optimisation (PPO) algorithm, as implemented in the Stable-Baselines3 library (Raffin et al., 2021). The PPO algorithm (Schulman et al., 2017) is a well-established, on-policy, model-free RL algorithm known for its robustness and effectiveness across various domains (OpenAI et al., 2019). PPO is an on-policy algorithm that iteratively improves a policy network by learning from batches of experience. The crucial link between our reward function (10) and the policy update is the advantage function. Rewards, (r_t), are used to compute an advantage estimate, $\hat{A}_t$, for each action taken. This value represents how much better or worse that action was compared to the policy's average performance in that state. The PPO algorithm then optimizes a composite loss function, which can be conceptually summarized as:

$$L_{total} = L_{policy}(\hat{A}_t) + c_1 \cdot L_{value} + c_2 \cdot S_{entropy} \tag{11}$$

Here, the policy loss, L_{policy}, uses the advantage estimates $\hat{A}_t$ to encourage actions that led to positive outcomes. The value loss, L_{value}, trains a separate network head to predict the expected cumulative reward from any given state. Finally, the entropy bonus, $S_{entropy}$, encourages exploration. The coefficients c_1 and c_2 are weighting hyperparameters. In this way, the rewards our agent receives directly shape the advantage calculations that, in turn, guide the policy towards achieving the goal of minimizing lap time.

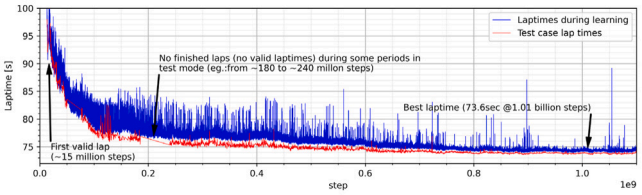

Fig. 3. Learning curve of the training, showing how laptimes of the Agent improved during the training. Exploited Laptimes corresponds to test laps, where deterministic NN outputs (mean values) were used as actions.

To accelerate training, experiences were gathered in parallel using 24 instances of the TORCS environment running simultaneously. Training sessions were conducted on DELL R730 E5-2670 v3 hardware, running for a maximum of 10 days (approximately one billion environment steps). Agent performance was monitored throughout training, with evaluation episodes performed every 10,000 steps using the deterministic mean of the policy network's output actions (whereas actions were sampled stochastically during training). Key hyperparameters specific to this study are detailed in Table A.3 in the Appendix; for parameters not listed there, the Stable Baselines3 defaults were utilized, as documented in Table A.5.

4. Simulation results

Fig. 3 shows the laptimes as a function of training steps. The blue curve shows the laptimes of the finished episodes during training, while the red curve shows the laptimes during evaluation episodes (test case lap times). Interestingly, although the agent learns to drive around the track after only ~15 million steps, it takes an additional ~billion steps to improve ~2 s in lap times. This may reflect the high sample complexity required for meticulously fine-tuning control strategies at the absolute limit of adhesion. It is also worth noting that there are periods at the beginning of the training where the agent cannot finish a lap in test mode after it has already learned it.

Figures in this chapter have a similar overall structure. In Fig. 4 the graphs use the distance traveled along the track centerline as the horizontal axis. The vertical axes show "driver inputs", as Steering Wheel Angle (SWA) on top and throttle/brake signals (+1 means full throttle, −1 means full brake pedal application) below. The next graph then shows vehicle speed with wheel speeds in [m/s] (y axis is in offset), and the last graph shows the track position (basically the racing line), where +1 refers to the left, and −1 the right side limit of the racetrack. There is also a "patch" in the middle of the speed trace. This patch is created by plotting the longitudinal and lateral acceleration of the vehicle. This plot is referred to as the "GG diagram" in racing terms. A GG diagram that is "round" or "fat" usually indicates that the car is being driven close to its limits, while a GG diagram that is "cross-like" or "thin" usually suggests a more road-car-like driving style, where longitudinal and lateral accelerations do not tend to mix. To help comprehend the track layout, the plots also feature the vehicle's traveled trajectory as an overlayed single line on the steering, throttle/brake and speed diagrams.

A lap driven by a professional e-Sport driver is depicted in Fig. 4. Gergely Baldi won the 2020 E-sport WTCR, which also featured professional "real" touring car champions. He spent ~18 h to make the lap shown in this figure. Although sim racers can easily spend 20–30 h to really improve their laps, this is already a decent champion level lap. The SWA trace shows an Understeer (US) vehicle behavior. There are several cues for this. On the one hand, it is a relatively smooth trace. An Oversteer (OS) car usually is more unstable, which usually needs more corrections on the steering and thus results in a less smooth

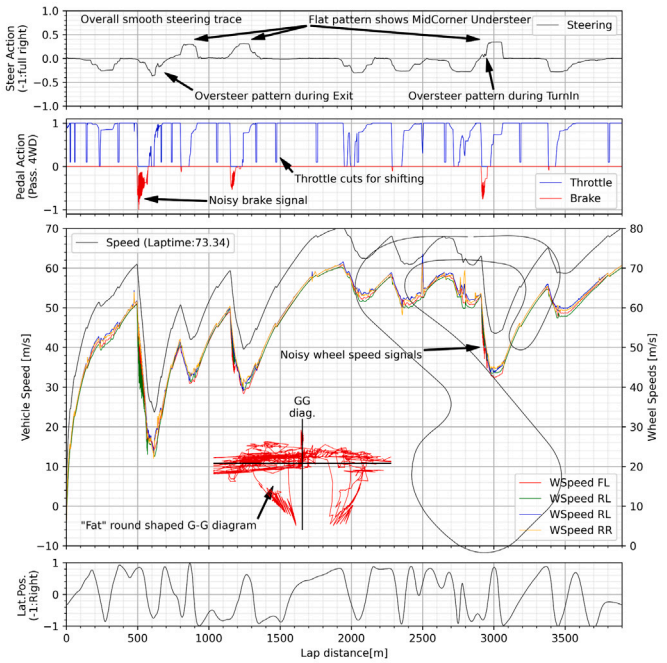

Fig. 4. Data plots from the human champion's lap, illustrating key driving characteristics. The steering trace shows a strategy to manage understeer (smooth plateaus) with sharp corrections for throttle-induced oversteer on exits. The "round" shape of the GG-plot and the full use of the track width confirm that the vehicle was consistently operated at its grip limits. Pedal inputs show artifacts such as throttle cuts for gear shifts and noise in the brake signal.

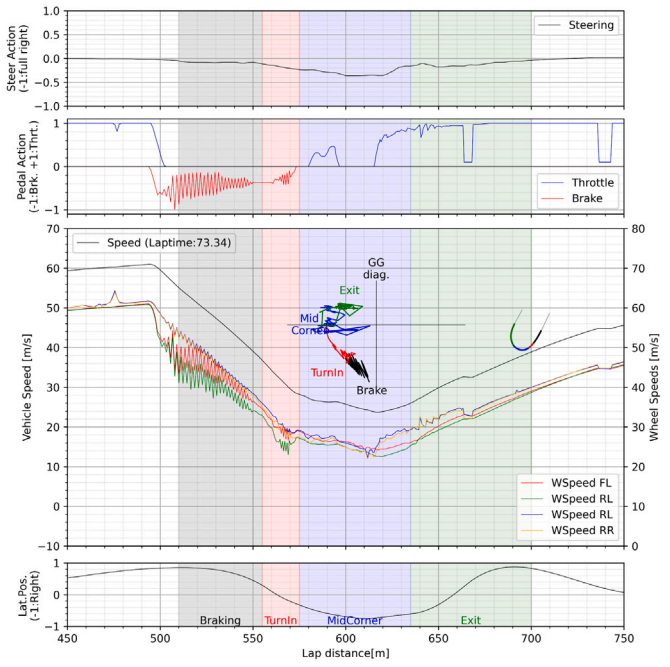

Fig. 5. Detailed analysis of the human champion's driving in Turn2. The top plot reveals a counter-steer correction during the exit phase to manage oversteer. The pedal action plot below it highlights the high-frequency noise in the brake signal, likely from the simulator hardware.

trace. Also, it can be seen that the maximum Steering Wheel Angle (SWA) values in each corner show a "flat', constant value for some period around its maximum. This shows that the driver keeps a given constant steering angle around the mid-corner, which usually means it is waiting for the car to rotate, and this "under-rotation" is also an US vehicle property. Having cars with US behavior is normal in simulator games such as TORCS. Vehicle control is much harder in simulation than in reality because there is less feedback from the car. Speed and distance are more difficult to estimate on the screen compared to real life, and vibrations, accelerations, and forces are not felt in simulation. For this reason, basic car setups in games tend to go towards US, as such a vehicle tends to be more stable and easier to control. Although sim racers tends to tune their car setups, in this work this was not allowed, both the Agent and the Human driver used the same car and setups.

OS behavior can be seen only in some places. For Example, during corner exit, in Turn2, the steering value does not decrease smoothly but rather quickly, with some correction. This shows that the driver forces the car to oversteer with "kicking" the throttle, which results in steering angle correction. This driver behavior is the result of the vehicle's inherent US nature. The driver need to apply this strategy only to make the car "over-rotate" after experiencing "under-rotation" earlier in mid-conrer. Corner entry OS can be noticed only in 2nd to last corner in case of this Human driven lap. This corner closely follows a quick right turn sequence, and therefore it is necessary to start braking right in the middle of a right-to-left steering motion. This makes the car unstable, and hence OS.

Considering the Throttle/Brake trace, two unexpected patterns can be noted. On the one hand, there is a strange noise in braking (most obvious in Turn2) and there are some drops (gear cuts) in the throttle in straight lines. Gear cuts are present in case of the Human driver but not for the Agent. The human driver preferred to use manual shifts, while in case the Agent a simple automatic gear shift logic was used as Agent inputs were designed to be only the throttle/brake signal. This setup was chosen to ensure the benchmark data was of the highest possible fidelity, by allowing the e-sport champion to perform in a familiar configuration and use his preferred transmission choice. Although it is fairly straightforward to understand the throttle cuts that come from gear shifts, the reason for the brake noise is not trivial.

Fig. 5 shows Turn2 zoomed in for better understanding the unusual brake signal. The main phases of the corner (Braking - TurnIn - Mid-Corner - Exit) are color-wise highlighted. There is clear high-frequency noise in the brake pedal signal, that results in some high-frequency in the wheel speed traces too. The most likely explanation for this is the error of the brake pedal potentiometer in the Human Driver's simulator equipment. It was not possible to overcome this problem under the scope of this work, but it seemed obsolete, too. Interviewing the driver made clear that he was not aware of this issue and also felt that this did not affect his performance.

Fig. 6 shows a lap driven by the trained Agent. The only trace that is significantly different from the Human driver is the steering wheel angle trace, as the Agent applies much harsher steering inputs. This is to be expected, as the agent in this work was not subject to any rate limit, and the power needed to turn the steering wheel was not modeled. Considering throttle/brake signal the pattern here is really close to Human driver trace. In a straight line there is full throttle application (flat sequences at maximum (+1)) When the agent reaches the braking point before a given corner, this maximum throttle turns into maximum brake signal (maximum, meaning maximum brake without locking the wheels).

The wheel speed signals in Turn2 show an interesting pattern. Fig. 7 shows that the agent uses the maximum brake command and locks the wheels. This is strange, as in other corners the brake signal shows that the agent did learn to brake at the tire grip limit. The steering trace shows a sudden turn in the opposite direction to the corner at the start of steering (~520 m) followed by an oversteered pattern, resulting in drift, and missing the apex. Although this behavior in this corner was not seen in most of the training runs, the lap presented here was the

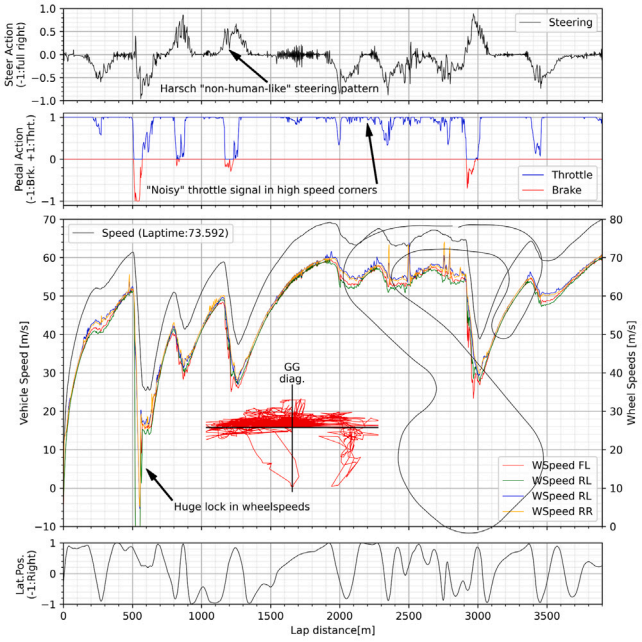

Fig. 6. Analysis of the Agent's best lap. The top plot shows exceptionally sharp steering actions, a behavior enabled by the simulation's lack of actuator rate limits. Below, the wheel speed data clearly indicates a major wheel-locking event during the braking phase for Turn2.

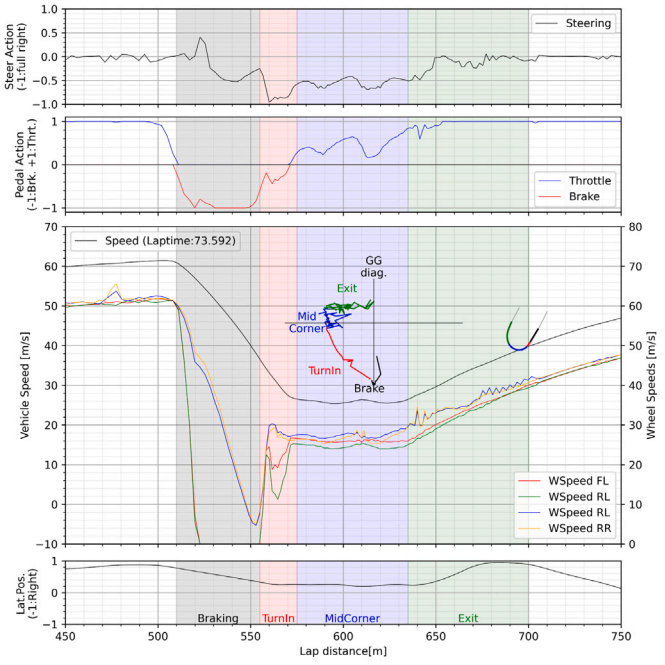

Fig. 7. Detailed analysis of the agent's suboptimal behavior in Turn2. The wheel speed traces reveal a complete wheel lock under maximum braking. Concurrently, the steering plot shows a sharp, counter-intuitive input at turn-in, followed by a large oversteer correction, which results in a missed apex and significant time loss.

fastest among all (meaning other parts of the lap were slower in other cases). It should be stated that this is not an optimal behavior in Turn2. This is proven by the time difference graph at the bottom of Fig. 9, which shows a sudden jump of ~0.5 s, meaning that the Agent loses

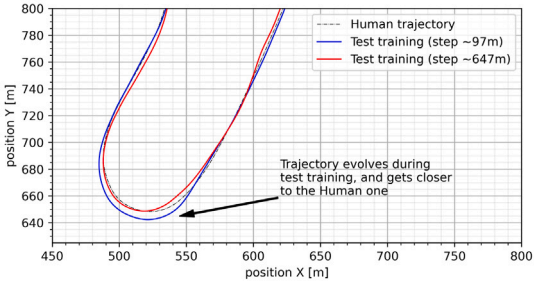

Fig. 8. Evolution of the agent's racing line in Turn2 during a dedicated test training. The plot compares the trajectory from an early stage of training with a much later one. The later trajectory is visibly tighter and more closely matches the optimal line (dashed human trajectory), demonstrating the policy's refinement over training and its ability to correct the initial suboptimal approach.

here a lot compared to the Human. It is difficult for the Agent to learn a proper braking behavior. First, it learns to drive as fast as possible in a straight line, so it naturally starts by overshooting the first corner where braking would be necessary (Turn2 in this case). Although the Agent does learn to brake eventually, it takes several million steps until it really finds the proper braking strategy.

Fig. 8 shows two trajectories. To investigate further the Agents" suboptimal behavior in Turn2, a dedicated test training was performed. In this training, the episodes were shorter, as they ended after the 2nd corner. In this way, in these short episodes, more experience was gathered in this specific corner (Turn2) with the same number of training steps. In Fig. 8 one line shows a trajectory from the early part of the training (~97 million steps), while the other comes from much higher training steps (~647 million steps). It can be seen that the early trajectory is "wider" just like the original line, and as the test training progresses, the later trajectory becomes closer to the racing line used by the Human champion. Based on these results, it seems clear that with more compute resources (and probably some more hyper-parameter tuning), this behavior can be avoided and even better laptimes can be achieved. It is also noteworthy that the Human driver reported that the inside of that corner is bumpy, which makes the usual usage of curbs (and the inside edge of the track) hurt laptimes, which could also add difficulties to the Agent to learn this corner.

Fig. 9 shows a comparison between the Human champion and the Agent. There are some general differences that can be noted in this Figure. In the bottom the time difference channel can be seen, that shows the time advantage of the Agent to the Human. Negative values show that the Human reached a given distance in less time than the Agent. Big changes in this graph indicate places where lap time is won/lost compared to each other. In the top graph, the earlier mentioned "un-humanly" harsh steering pattern of the Agent is much more obvious overlayed to the Human one. The throttle/brake trace overlay reveals, that the Human driver creates a small "chamfer" or "notch" -like pattern before full throttle applications. This was not so obvious when the Human lap was analyzed on its own, but using the Agents lap helped to highlight this pattern. This shows that the Human waits in most corners just before applying full throttle, that coincides with the plateau in the Steering trace. This reinforces our observation that the Human is "waiting" in this phase for the US car to rotate, while the Agent's driving approach (steering, braking, throttle all combined) makes possible a sudden full-throttle application during corner exits. Analyzing the braking behavior, it becomes evident that the Agent usually opts for later brake points compared to the human, even at the cost of exit performance. This tendency is a direct consequence of the discounted reward structure. The agent learns easier that aggressive braking yields a large, immediate "progress" reward, the value of which

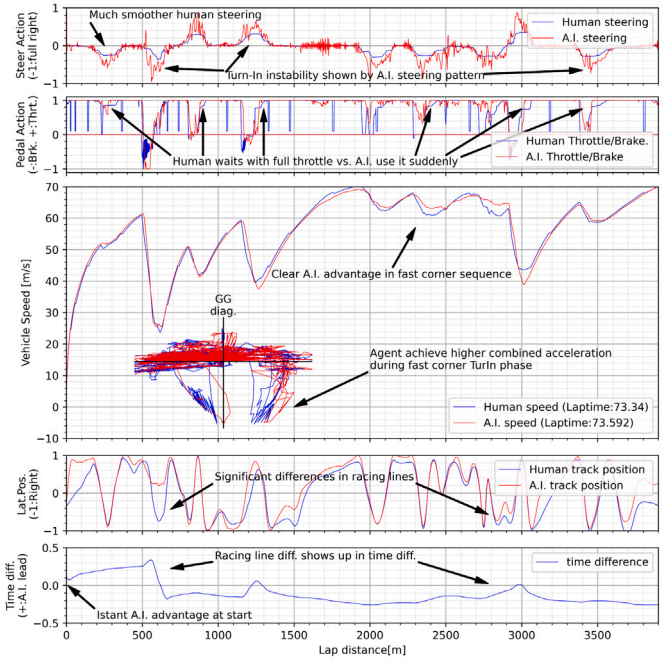

Fig. 9. Comparison of the full lap between the Agent and the Human champion, revealing distinct strategies: the human uses smooth inputs, and manage understeer with throttle "kick", while the agent use harsh inputs and exploit the simulation's modeling limits. The lateral position shows differences in racing lines only in the difficult corners. The bottom plot quantifies the time difference, highlighting sections where each driver gains an advantage, such as the agent's gain in the high-speed sequence.

Table 1
Quantitative comparison of sector and total lap times for the best-performing laps of the human champion and the DRL agent. Time differences are calculated as Human time - Agent time, where a positive value indicates the agent was faster in that sector.

Sector splits	Human time (s)	Agent time (s)	Time diff. (s) (+): A.I. faster
500 m	12.304	12.069	+0.235
2000 m	30.012	30.523	−0.511
2900 m	13.898	13.711	+0.187
Finish	17.106	17.289	−0.183
Total lap time	73.320	73.592	−0.272

is discounted less than the potential reward from a faster exit further in the future. While this strategy ultimately leads to a competitive lap time, it is a hallmark of the myopic nature of the dense reward signal. Although this behavior is expected to disappear with continued training, it suggests that additional hyperparameter tuning, additional training iterations, reward shaping, and more sample efficient methods might further enhance lap times.

As shown in Table 1, a quantitative analysis of the sector times reveals the nuanced differences in performance. The agent establishes a clear advantage of +0.235 s in the first 500 m, capitalizing on its optimized start. However, it loses over half a second in the next section (until ~2000 m) due to its suboptimal approach into Turn2. The agent regains +0.187 s through the high-speed cornering sequence (until ~2900 m) before finally losing time again in the last sector, which includes the technically difficult braking zone for the second-to-last corner. While the final lap times are close, this breakdown illustrates the contrasting strengths and weaknesses of the two driving styles.

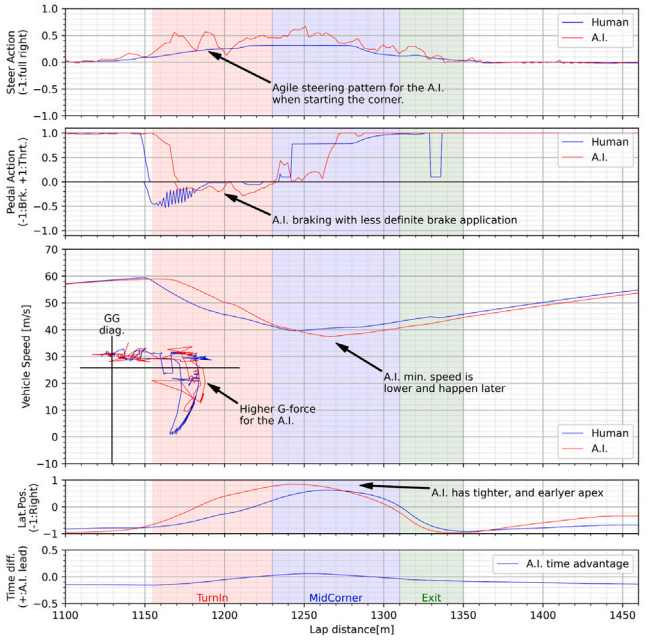

Fig. 10. Analysis of the agent's aggressive entry strategy in Turn4. The agent's less definitive braking at higher lateral G-force (visible in the GG-plot) induces instability, requiring sharp steering corrections. This results in an earlier apex and a later minimum speed compared to the human, showcasing a classic 'entry overshoot' that costs time on corner exit.

To illustrate the mentioned "braking/entry overshoot" behavior, Fig. 10 provides a detailed view of Turn4. In this corner, the difference between entry speed and minimum speed is minimal, and the required short, soft braking occurs during lateral load. This means there is no straight-line braking phase, so the initial phase of this corner is the TurnIn phase. The GG diagram shows that the Human uses more pronounced braking action, achieving a different direction on the GG (more longitudinal - less lateral acceleration). The Agent brakes at higher lateral acceleration with less definite braking (lower brake values) and achieves higher combined accelerations, that pushes the car in unstable (OS) behavior. The steering trace indicates that the Agent compensates for the unstable vehicle behavior with vigorous steering, proving the overly aggressive turn-in approach. The Agent takes an 'early apex' (around 1240 m), reaching the innermost part of the corner before the Human (around 1260 m). Notably, the Agent's minimum speed occurs approximately 20 m after the apex, whereas for the Human, the minimum speed is reached about 20 m before the apex. This indicates again, an 'overshoot' in the TurnIn phase and as a result, in the MidCorner and Exit phases, the Human car is faster, causing the Agent to lose the advantage it gained during braking showed in the time-difference graph.

Although Turn4 is one of the three places where the race line between the agent and the human is slightly different, it is interesting to see that in general the Agent uses a very similar racing line to the Human. This is shown by the racing line (2nd to bottom graph in Fig. 9). Finding an ideal racing line that is close to a human professional is a great result and a testament to the opportunities presented by the RL framework. Other than Turn4 there are two more places where significant differences can be seen in the taken lines. A key finding of our race-engineering style analysis is that the agent's struggles are concentrated in the exact same "tricky" corners that also challenge the human champion. This parallel is particularly evident in the second-to-last corner, a left turn that immediately follows a quick right-hand sequence. Here, braking must occur during a rapid change of direction, a maneuver that destabilizes the car for both the human and the AI,

highlighting a shared challenge in mastering the vehicle's transient dynamics. The difficult nature of Turn2 was also analyzed already, where the Agent learned a strange, suboptimal behavior.

The start of the lap shows an unusual time difference. It shows an instant advantage for the agent. The reason for this is not obvious. To investigate, the first ~2 s of the episode is shown in Fig. 11. While both the Human and the Agent begin from identical stationary positions, speed, and distance graphs reveal that the human driver's vehicle remains motionless for approximately 0.15 s after start. This is due to fundamental differences in episode initializations. The agent's episodes start directly with first gear engaged at 0 RPM, enabling immediate full throttle application. Conversely, the human driver's session followed conventional racing game protocols requiring manual gear engagement after an initial countdown phase. Throttle traces confirm this operational difference - the agent initiates with maximum throttle (1.0) from t = 0 s, while the human driver's initial throttle application shows the gearshift-induced reductions (0.4–0.6 range). This procedural artifact creates an artificial ~0.1 s advantage for the agent during start, though subsequent sector times remain comparable. For future research, it would be beneficial to address and mitigate these differences; however, within the scope of this study, they were considered acceptable. Although the agent's initial time gain arises from procedural differences rather than learned skill, its steering behavior highlights a notable adaptation. To prepare for the first right-hand corner, both drivers must shift leftward on the track. The human driver accomplishes this transition gradually, as shown by smooth lateral and steering traces. Conversely, the agent makes an abrupt leftward turn immediately after starting and straighten the steering only when rear-wheel spin begins to destabilize the car. This maneuver exploits its initialization advantage — starting in first gear at 0 RPM — by using this brief stable phase to position itself effectively. Such behavior underscores the strength of reinforcement learning in discovering innovative strategies that capitalize on unique environmental conditions.

Another notable difference between the Agent's and the Human driving is how the Agent is at a clear advantage in the middle section, at the fast, bumby right-hander sequence (~2000 m–2700 m) where the car. The reason for this seems to be that the Agent has learned to control the unstable vehicle with a responsiveness potentially exceeding human limits, an advantage likely stemming directly from the unconstrained steering rate in the simulation. This "unfair" advantage can be mitigated by modeling the missing rate limit on the steering for future work.

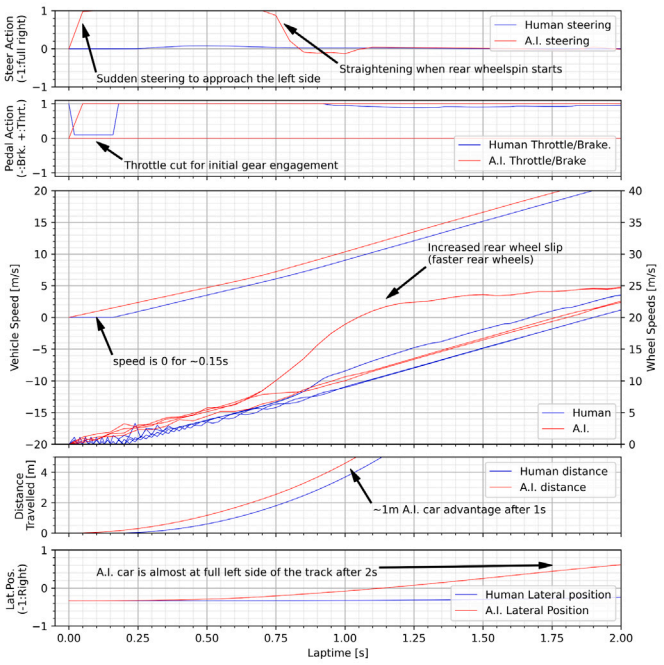

Fig. 11. Analysis of the start, highlighting a procedural advantage for the agent, where the human remains stationary for ~0.15 s due to manual gear engagement. The agent also uses this initial phase to execute a sharp, immediate turn to the left side of the track, positioning itself for the first corner more aggressively than the human driver.

5. Conclusion

This research successfully demonstrated the application of Deep Reinforcement Learning (DRL), specifically the Proximal Policy Optimisation (PPO) algorithm, to train an autonomous agent capable of driving a simulated race car at the grip limit, achieving lap times comparable to a human e-sport world champion. By formulating the complex task of time-optimal racing as a DRL problem utilizing real vehicle friendly state inputs, we showed that an end-to-end learning approach can effectively discover sophisticated control strategies without explicit trajectory planning or grip estimation subsystems, which often pose challenges in high-performance driving scenarios.

A key contribution of this work is the validation of the commonly used "progress reward" by demonstrating its direct relationship to the time-difference feedback mechanisms employed by professional human drivers during practice. This provides a human-centric justification for this reward structure in the context of autonomous racing. The trained agent achieved performance on par with the human champion, showcasing the potential of DRL in this domain. However, detailed analysis revealed significant differences in driving style. The AI agent learned to exploit the specifics of the simulation environment, particularly the absence of a steering rate limit, adopting an aggressive strategy that induced oversteer to counteract the vehicle's inherent understeer characteristics. Conversely, the human champion exhibited smoother inputs, managing the understeer more traditionally and displaying nuanced throttle control during corner exits. These contrasting approaches, while yielding similar overall lap times, highlight how AI may find alternative, sometimes non-human-like, optima within the given physical constraints and control freedoms.

The analysis also identified specific areas for improvement and limitations inherent in the current setup. The agent's suboptimal or overly aggressive behavior in certain challenging corners (like Turn2 and Turn4), potentially due to insufficient exploration during training or simulation artifacts like track bumpiness, suggests that further training or refined reward shaping could yield even better performance. Notably, our analysis reveals that these struggles mirror the challenges faced by the human driver in the same technically demanding sections, making this Human-AI parallel a significant finding of our work. The tendency for "entry overshoot" is a direct consequence of the discounted reward structure, which incentivizes large, immediate rewards from late braking. Furthermore, the requirement of approximately one billion environment steps for training highlights a significant sample inefficiency. This makes the direct application of this method to real-world systems impractical without leveraging sim-to-real transfer techniques or exploring more sample-efficient algorithms. Several experimental asymmetries must be acknowledged as limitations that create an uneven comparison ground. The agent benefited from an unconstrained steering rate, a simplified automatic transmission, and a procedural start advantage. The latter two were deliberate methodological choices made to prioritize the fidelity of the human benchmark data. Although we isolated and accounted for these artifacts in our analysis, they represent idealizations that will be fully addressed in future work. It is also important to note that the training protocol was intentionally designed to create a specialist agent for this specific track and vehicle to allow for a deep, micro-level performance analysis; as such, the current agent is not expected to generalize, and a rigorous evaluation of the method's robustness across

different settings remains an essential direction for future research. It is critical to note that the agent's ability to apply steering inputs at a 20 Hz frequency without rate limitations provides a distinct advantage over the human driver, whose effective control frequency is closer to 10 Hz. This "superhuman" capability, particularly evident in the agent's superior performance in the fast chicane section, is a key factor in its overall lap time and represents a significant idealization in the current simulation.

Furthermore, it is crucial to acknowledge that this study's validation is exclusively within a simulated environment. We frame this work as a foundational step, intended to first confirm that the DRL methodology can achieve expert-level performance before tackling the significant cost and safety challenges of real-world deployment. Our experimental design was deliberately chosen to serve as a bridge towards this goal; by using only "real-world-friendly" vehicle dynamic state inputs, our approach is more directly transferable than methods relying on privileged simulation information. However, the sim-to-real gap remains a substantial hurdle, and future work must focus on validating these learned policies on physical platforms.

Future research should focus on bridging this gap between simulation and reality. Implementing realistic actuator constraints, such as steering rate and force limits, would enable a fairer comparison with human capabilities and potentially lead the agent to discover more human-like strategies. A critical next step, as highlighted by our analysis, is to validate the sim-to-real transferability of these policies, for example by using hardware-in-the-loop (HIL) test benches, small-scale autonomous platforms, or higher-fidelity simulation environments before the final step that is the physical deployment in a real race car. To address sample inefficiency, future work should also explore off-policy algorithms like Soft Actor-Critic (SAC) or model-based methods like Dreamer (Hafner et al., 2020), which have demonstrated greater data efficiency in other domains. Extending the agent's control to include independent four-wheel torque vectoring and steering could unlock further performance gains, mirroring advanced real-world vehicle control systems. Investigating the agent's interaction with driver aids like Anti Block System (ABS) and Traction Control System (TCS) would be crucial for applications aiming towards production vehicles. Finally, validating these learned policies on physical platforms, remains a vital next step towards leveraging these high-performance simulated agents for enhancing the safety and performance of real-world autonomous driving systems, particularly in critical emergency maneuvers that push vehicles to their handling limits.

CRediT authorship contribution statement

Gergely Bári: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **László Palkovics:** Supervision, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix. Training parameters

See Tables A.2–A.5.

Table A.2
Parameters for the reward function in (Eq. (10)).

Parameter	Value
Reference speed (v^{ref})	20
Action scaling (p^{sc})	15
Bound for scaled action (p^{bnd})	1.2
Termination reward components (r_t^{ter}):	
Reaching the finish (`dist_episode > 3900`)	+100
Leaving the track (`|track_pos| > 1.2`)	−10
Turning back (`angle < 0`)	−10
Damage the car (`damage > 0`)	−10
Progress backwards (`speed < 0`)	−10
Low prog. (`timestep > 500 AND ep_reward < 0`)	−10

Table A.3
Training parameters for Stable Baselines PPO, and the TORCS simulator environment. PPO parameters not listed, are default as per StableBaselines3 documentation.

Parameter	Value
Learning rate - with decay	$[1 : 2.5, 0 : 0.5] \cdot 10^{-4}$
Maximum training steps	$1.5 \cdot 10^9$
Discount factor	0.995
Entropy coefficient	0.01
Value function coefficient	0.5
Policy clip range	0.2
Value function clip range	0.2
Environment instances	24
Batch size	512
Agent-to-Environment timestep (ts)	0.05 s
Used track name	Brondehach
Used car model name	Car7
`ASR_ON`	False
`ASR_ON`	False
Timestep in physics simulation	0.002 s

Table A.4
Observation vector elements.

Measure	Description	Scale
`LiDaR`	17 ray, in-plane lidar signal [m]	/300
`Whl.Speeds`	4 wheel speed signals [m/s]	/80
`Epis. dist.`	Distance along the centerline [m]	
`Angle`	Car x axis angle to track CL [rad]	$/\pi$
`Speed x`	CG longitudinal speed [m/s]	/300
`Speed y`	CG lateral speed [m/s]	/300
`Yawrate`	Yaw rate [rad/s]	$/\pi$
`x Accel.`	Longitudinal acceleration [m/ss]	/100
`y Accel.`	Lateral acceleration [m/ss]	/100

Table A.5
Default Stable Baselines3 PPO hyperparameters.

Parameter	Default value
`n_steps`	2048
`n_epochs`	10
`gae_lambda`	0.95
`normalize_advantage`	True
`max_grad_norm`	0.5
`use_sde`	False
`sde_sample_freq`	−1
`target_kl`	None

References

Acosta, M., Kanarachos, S., 2018. Tire lateral force estimation and grip potential identification using neural networks, extended Kalman filter, and recursive least squares. Neural Comput. Appl. 30, 3445–3465.

Bári, G., 2018. Modelling the situation of driving on the grip limit with DDPG algorithm. IOP Conf. Ser.: Mater. Sci. Eng. 393, 012031. http://dx.doi.org/10.1088/1757-899X/393/1/012031, URL: https://iopscience.iop.org/article/10.1088/1757-899X/393/1/012031.

Bári, G., Palkovics, L., 2026. Vision based driving agent for race car simulation environments. In: Conte, G.L., Sename, O. (Eds.), Proceedings of 12th International

Conference on Mechatronics and Control Engineering. Springer Nature, Singapore, pp. 177–188. http://dx.doi.org/10.1007/978-981-96-6452-8_15.

Brown, M., Gerdes, J.C., 2020. Coordinating tire forces to avoid obstacles using nonlinear model predictive control. IEEE Trans. Intell. Veh. 5, 21–31. http://dx.doi.org/10.1109/TIV.2019.2955362, URL: https://ieeexplore.ieee.org/document/8910387/?arnumber=8910387. conference Name: IEEE Transactions on Intelligent Vehicles.

Brunnbauer, A., Berducci, L., Brandstätter, A., Lechner, M., Hasani, R., Rus, D., Grosu, R., 2022. Latent imagination facilitates zero-shot transfer in autonomous racing. http://dx.doi.org/10.48550/arXiv.2103.04909, URL: http://arxiv.org/abs/2103.04909. arXiv:2103.04909 [cs].

Cataffo, V., Silano, G., Iannelli, L., Puig, V., Glielmo, L., 2022. A nonlinear model predictive control strategy for autonomous racing of scale vehicles. In: 2022 IEEE International Conference on Systems, Man, and Cybernetics. SMC, IEEE, Prague, Czech Republic, pp. 100–105. http://dx.doi.org/10.1109/SMC53654.2022.9945279, URL: https://ieeexplore.ieee.org/document/9945279/.

Fuchs, F., Song, Y., Kaufmann, E., Scaramuzza, D., Dürr, P., 2021. Super-human performance in gran turismo sport using deep reinforcement learning. IEEE Robot. Autom. Lett. 6, 4257–4264. http://dx.doi.org/10.1109/LRA.2021.3064284, URL: https://ieeexplore.ieee.org/document/9372847.

Haarnoja, T., Zhou, A., Abbeel, P., Levine, S., 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In: Proceedings of the 35th International Conference on Machine Learning. PMLR, pp. 1861–1870, URL: https://proceedings.mlr.press/v80/haarnoja18b.html.

Hafner, D., Lillicrap, T., Ba, J., Norouzi, M., 2020. Dream to control: Learning behaviors by latent imagination. http://dx.doi.org/10.48550/arXiv.1912.01603, URL: http://arxiv.org/abs/1912.01603. arXiv:1912.01603 [cs].

Hafner, D., Lillicrap, T., Fischer, I., Villegas, R., Ha, D., Lee, H., Davidson, J., 2019. Learning latent dynamics for planning from pixels. URL: http://arxiv.org/abs/1811.04551. arXiv:1811.04551 [cs, stat].

Hafner, D., Pasukonis, J., Ba, J., Lillicrap, T., 2025. Mastering diverse control tasks through world models. Nature 640, 647–653. http://dx.doi.org/10.1038/s41586-025-08744-2, URL: https://www.nature.com/articles/s41586-025-08744-2.

Kelly, D.P., Sharp, R.S., 2010. Time-optimal control of the race car: a numerical method to emulate the ideal driver. Veh. Syst. Dyn. 48, 1461–1474. http://dx.doi.org/10.1080/00423110903514236.

Lee, H., Seno, T., Tai, J.J., Subramanian, K., Kawamoto, K., Stone, P., Wurman, P.R., 2025. A champion-level vision-based reinforcement learning agent for competitive racing in gran turismo 7. IEEE Robot. Autom. Lett. 10, 5545–5552. http://dx.doi.org/10.1109/LRA.2025.3560873, URL: https://ieeexplore.ieee.org/abstract/document/10964853.

Nashashibi, F., 2018. End-to-end race driving with deep reinforcement learning. In: 2018 IEEE International Conference on Robotics and Automation. ICRA, URL: https://www.academia.edu/80293831/End_to_End_Race_Driving_with_Deep_Reinforcement_Learning.

OpenAI, Berner, C., Brockman, G., Chan, B., Cheung, V., Dębiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C., Józefowicz, R., Gray, S., Olsson, C., Pachocki, J., Petrov, M., Pinto, H.P.d.O., Raiman, J., Salimans, T., Schlatter, J., Schneider, J., Sidor, S., Sutskever, I., Tang, J., Wolski, F., Zhang, S., 2019. Dota 2 with large scale deep reinforcement learning. http://dx.doi.org/10.48550/arXiv.1912.06680, URL: http://arxiv.org/abs/1912.06680. arXiv:1912.06680 [cs].

OpenSource, 2020. TORCS - the open racing car simulator. URL: https://sourceforge.net/projects/torcs/.

Pedone, S., Fagiolini, A., 2023. Robust discrete-time lateral control of racecars by unknown input observers. IEEE Trans. Control Syst. Technol. 31, 1418–1426. http://dx.doi.org/10.1109/TCST.2022.3214054, URL: https://ieeexplore.ieee.org/document/9928032. conference Name: IEEE Transactions on Control Systems Technology.

Raffin, A., Hill, A., Gleave, A., Kanervisto, A., Ernestus, M., Dormann, N., 2021. Stable-baselines3: Reliable reinforcement learning implementations. J. Mach. Learn. Res. 22, 1–8, URL: http://jmlr.org/papers/v22/20-1364.html.

Remonda, A., Hansen, N., Raji, A., Musiu, N., Bertogna, M., Veas, E., Wang, X., 2024. A simulation benchmark for autonomous racing with large-scale human data. http://dx.doi.org/10.48550/arXiv.2407.16680, URL: http://arxiv.org/abs/2407.16680. arXiv:2407.16680.

Remonda, A., Krebs, S., Veas, E., Luzhnica, G., Kern, R., 2022. Formula RL: Deep reinforcement learning for autonomous racing using telemetry data. http://dx.doi.org/10.48550/arXiv.2104.11106, URL: http://arxiv.org/abs/2104.11106. arXiv:2104.11106.

Remonda, A., Veas, E., Luzhnica, G., 2021. Comparing driving behavior of humans and autonomous driving in a professional racing simulator. PLOS ONE 16, e0245320. http://dx.doi.org/10.1371/journal.pone.0245320, URL: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0245320.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O., 2017. Proximal policy optimization algorithms. http://dx.doi.org/10.48550/arXiv.1707.06347, URL: http://arxiv.org/abs/1707.06347. arXiv:1707.06347 [cs].

Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., Hassabis, D., 2016. Mastering the game of Go with deep neural networks and tree search. Nature 529, 484–489. http://dx.doi.org/10.1038/nature16961, URL: https://www.nature.com/articles/nature16961. number: 7587.

Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., Riedmiller, M., 2014. Deterministic policy gradient algorithms. In: Proceedings of the 31st International Conference on Machine Learning. PMLR, pp. 387–395, URL: https://proceedings.mlr.press/v32/silver14.html.

Vasco, M., Seno, T., Kawamoto, K., Subramanian, K., Wurman, P.R., Stone, P., 2024. A super-human vision-based reinforcement learning agent for autonomous racing in gran turismo. http://dx.doi.org/10.48550/arXiv.2406.12563, URL: http://arxiv.org/abs/2406.12563. arXiv:2406.12563 [cs].

Venkatesh, P., Rana, R., PM, H., 2021. Fast and real-time end to end control in autonomous racing cars through representation learning. http://dx.doi.org/10.48550/ARXIV.2111.15343, URL: https://arxiv.org/abs/2111.15343. version Number: 1.

Verschueren, R., De Bruyne, S., Zanon, M., Frasch, J.V., Diehl, M., 2014. Towards time-optimal race car driving using nonlinear MPC in real-time. In: 53rd IEEE Conference on Decision and Control. IEEE, Los Angeles, CA, USA, pp. 2505–2510. http://dx.doi.org/10.1109/CDC.2014.7039771, URL: http://ieeexplore.ieee.org/document/7039771/.

Wurman, P.R., Barrett, S., Kawamoto, K., MacGlashan, J., Subramanian, K., Walsh, T.J., Capobianco, R., Devlic, A., Eckert, F., Fuchs, F., Gilpin, L., Khandelwal, P., Kompella, V., Lin, H., MacAlpine, P., Oller, D., Seno, T., Sherstan, C., Thomure, M.D., Aghabozorgi, H., Barrett, L., Douglas, R., Whitehead, D., Dürr, P., Stone, P., Spranger, M., Kitano, H., 2022. Outracing champion Gran Turismo drivers with deep reinforcement learning. Nature 602, 223–228. http://dx.doi.org/10.1038/s41586-021-04357-7, URL: https://www.nature.com/articles/s41586-021-04357-7.

Zhu, Q., Schmid, M., Prucka, R., Boncimino, A., Paredis, C., 2024. Control informed design of the IAC autonomous racecar for operation at the dynamic envelope. http://dx.doi.org/10.48550/arXiv.2407.17737, URL: http://arxiv.org/abs/2407.17737. arXiv:2407.17737 [eess].