This paper has been accepted for publication at the
IEEE International Conference on Robotics and Automation (ICRA), Xi'an, 2021. ©IEEE

# Autonomous Overtaking in Gran Turismo Sport Using Curriculum Reinforcement Learning

Yunlong Song*, HaoChih Lin*, Elia Kaufmann, Peter Dürr, and Davide Scaramuzza

***Abstract*— Professional race-car drivers can execute extreme overtaking maneuvers. However, existing algorithms for autonomous overtaking either rely on simplified assumptions about the vehicle dynamics or try to solve expensive trajectory-optimization problems online. When the vehicle approaches its physical limits, existing model-based controllers struggle to handle highly nonlinear dynamics, and cannot leverage the large volume of data generated by simulation or real-world driving. To circumvent these limitations, we propose a new learning-based method to tackle the autonomous overtaking problem. We evaluate our approach in the popular car racing game Gran Turismo Sport, which is known for its detailed modeling of various cars and tracks. By leveraging curriculum learning, our approach leads to faster convergence as well as increased performance compared to vanilla reinforcement learning. As a result, the trained controller outperforms the built-in model-based game AI and achieves comparable overtaking performance with an experienced human driver.**

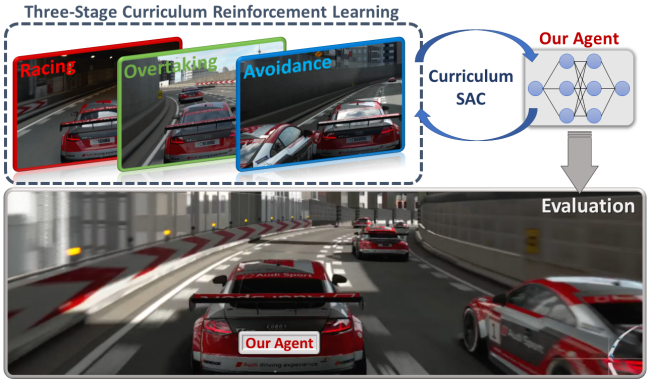


Fig. 1: A system overview of the proposed curriculum reinforcement learning method for addressing the autonomous overtaking problem in Gran Turismo Sport.

**Video:** `https://youtu.be/e8TVPv4D4O0`

## I. INTRODUCTION

The goal of autonomous overtaking in car racing is to overtake the opponents as fast as possible while avoiding collisions. Experienced race-car drivers can operate a vehicle at the limits of handling and, at the same time, perform overtaking during very extreme maneuvers. Developing an autonomous system that can achieve the same level of human control performance, or even go beyond, could not only shorten the travel time and reduce transportation costs but also avoid fatal accidents.

However, developing such an autonomous overtaking system is very challenging for several reasons: 1) The entire system, including the vehicle, the tire model, and the vehicle-road interaction, has highly complex nonlinear dynamics. 2) The intentions of other opponents are unknown, rendering most high-level trajectory planning algorithms incapable of reliably generating accurate overtaking trajectories. 3) The vehicle is already close to its physical limits, leaving very limited control authority for executing overtaking maneuvers.

Previous methods tackled the problem using classical trajectory generation and tracking techniques and relied on tools from dynamic modeling, optimal control, and nonlinear programming. Despite all the successes [1], this line of research has several limitations. For example, many trajectory planning algorithms [2]–[4] use a simplified vehicle model and neglect several real-world effects, such as the tire-road interaction and aerodynamic effects. These algorithms escalate in complexity when considering high-fidelity vehicle models and complex interactions among vehicles.

Recently, deep reinforcement learning (RL) has emerged as an effective approach in solving complex robotic control problems [5]–[7]. Particularly, model-free deep RL trains a parametric policy by directly interacting with the environment and does not assume knowledge of an exact mathematical model of the system, making the method well-suited for highly nonlinear systems and complex tasks. Furthermore, the neural network policies allow flexible controller design, allowing different state representations that range from high-dimensional images to low-dimensional states.

Here, we present a new learning-based system for high-speed autonomous overtaking. The key is to leverage task-specific curriculum RL and a novel reward formulation to train an end-to-end neural network controller. Our approach manifests faster convergence as well as increased performance compared to vanilla deep RL, which instead trains neural network policies directly for overtaking without any prior knowledge about driving. The proposed curriculum learning procedure can transfer the knowledge that is obtained from a single-car racing task to solve more complicated overtaking problems. As a result, our trained controller outperforms the built-in model-based controller and achieves comparable overtaking performance with an experienced human driver.

*These two authors contributed equally. Y. Song, E. Kaufmann, and D. Scaramuzza are with the Robotics and Perception Group, Dep. of Informatics, University of Zurich, and Dep. of Neuroinformatics, University of Zurich and ETH Zurich, Switzerland (`http://rpg.ifi.uzh.ch`). H. Lin and P. Dürr are with Sony AI Zurich. This work was supported by Sony AI Zurich, the National Centre of Competence in Research (NCCR) Robotics through the Swiss National Science Foundation, and the European Research Council Consolidator Grant (ERC-CoG) under the European Union's Horizon 2020 Research and Innovation Programme (Grant agreement No. 864042).

## II. Related Work

A bulk of research in autonomous driving has been focusing on developing safe overtaking systems in low-speed driving scenarios [1]. We categorize prior work in the domain of autonomous overtaking into two groups: model-based approaches and learning-based approaches.

*1) Model-based:* Model-based approaches attempt to tackle the problem via a modular architecture design, which breaks the overtaking problem down into trajectory planning and trajectory tracking. For instance, sampling-based trajectory planning methods [8], [9], such as Rapidly Exploring Random Trees (RRT), have been proposed for planning safe trajectories for autonomous overtaking. These methods generally make use of simplified vehicle models and basic vehicle kinematics. However, when a car operates close to the limits of handling and drives at high-speed, it is insufficient to ignore many real-world effects, such as tire-road interaction and aerodynamics effects introduced by the motion of other vehicles.

Optimization-based approaches, such as Model Predictive Control (MPC), are effective solutions to trajectory planning and tracking in autonomous overtaking [10]–[12], thanks to their capability of handling different constraints and robust performance against disturbances. Similar to motion planning algorithms, optimization-based approaches rely on the assumption of a simplified car model, and thus, do not guarantee that they can handle very complex nonlinear system dynamics. Besides, the requirement of solving nonlinear optimization online is computationally demanding for embedded systems.

*2) Learning-based:* Learning-based approaches, such as imitation learning [13], [14] and reinforcement learning [15]–[17], can in principle address the limitations of traditional modular and model-based approaches by learning parameterized policies that directly map sensory observations to control commands. One of the key advantages of using learning-based approaches is that they do not require perfect knowledge about the vehicle and its environment.

While imitation learning is an effective approach for training a neural network policy using experienced data demonstrated by human experts, overtaking is a sparse signal and can be difficult for human drivers. When deployed naively, imitation learning is sensitive to the distribution shift between the observations induced by the expert policy and the network policy. This problem can be alleviated using DAGGER [18], a time-consuming and expensive process for data collection.

Reinforcement learning (RL) seems to offer real potential for solving such complex decision-making problems by maximizing a reward signal that can formulate the overtaking problem properly. However, most advances in RL published to date are largely empirical. A thorough study on training methods and design choices in the autonomous overtaking domain is, unfortunately, lacking in the community. Our work is inspired by [6], but extends it to the more complex and challenging overtaking domain.

## III. METHODOLOGY

This section introduces the problem formulation of autonomous overtaking and its reward function design and describes how curriculum can be combined with off-policy actor-critic methods for training the neural network policy.

### A. Problem Formulation

High-speed overtaking in car racing involves two main objectives: minimizing the total overtaking time and avoiding collisions between the agent and other vehicles or obstacles. Intuitively, the agent that takes a short period to overtake its opponents needs to drive at high speed and has high collision probability, and vice versa. Hence, the optimal overtaking strategy defines the best trade-off between these two competing objectives.

*1) The Racing Problem:* We first consider a single-player racing problem, in which the goal is to drive a race car on a given track in minimum time. Instead of minimizing the time directly, the time-optimal objective is normally reformulated as minimizing the path of least curvature or the shortest path in order to use numerical optimization methods [19]. In [6], the authors propose a course-progress-proxy reward formulation, which closely represents the lap time and can be maximized using reinforcement learning. The course progress is determined by projecting the car's position to the point along the centerline (see Fig. 2). Hence, the racing reward $r_t^{\text{racing}}$ at the time stage $t$ is defined as:

$$r_t^{\text{racing}} = (cp(\boldsymbol{s}_t) - cp(\boldsymbol{s}_{t-1})) - c_w \rho_w |\boldsymbol{v}_t|^2 \tag{1}$$

where $cp(\boldsymbol{s}_t)$ is the centerline projection based on current car position $\boldsymbol{s}_t$. Here, $\rho_w$ is a binary flag indicating whether or not the wall collision occurs, and $c_w \geq 0$ is a hyperparameter for weighting the wall collision penalty. From the physical point of view, the first two terms encourage the learning policy to drive as fast as possible and the last term incentivizes to avoid the wall collision in the meantime. The last term is also depending on the collision kinetic, which is proportional to the square of the car's speed ($\boldsymbol{v}_t$).

*2) The Overtaking Problem:* The overtaking problem in car racing includes not only the objective of minimizing lap time, but also, avoiding collisions with other vehicles. We propose a novel continuous reward function ($r_t^{\text{overtaking}}$) for the overtaking problem. The formulation of the proposed continuous reward is expressed as:

$$\begin{aligned} r_t^{\text{overtaking}} &= r_t^{\text{racing}} - c_c \rho_c |\boldsymbol{v}_t|^2 + \\ &\sum_{\forall i \in C \setminus \{k\}} \left\{ \rho^i c_r [\Delta cp(s_{t-1}^i, s_{t-1}^k) - \Delta cp(s_t^i, s_t^k)] \right\} \end{aligned} \tag{2}$$

where,

$$\Delta cp(s_t^i, s_t^k) = cp(s_t^i) - cp(s_t^k)$$

$$\rho^i = \rho(s_t^i, s_t^k) = \begin{cases} 1, & |\Delta cp(s_t^i, s_t^k)| < c_d \\ 0, & \text{Otherwise} \end{cases}$$

where $C$ is a set of total simulated cars on the track, $k$ represents the ego-car controlled by the learning policy, $c_d$ is a hyperparameter for the detection range, $c_r$ is a

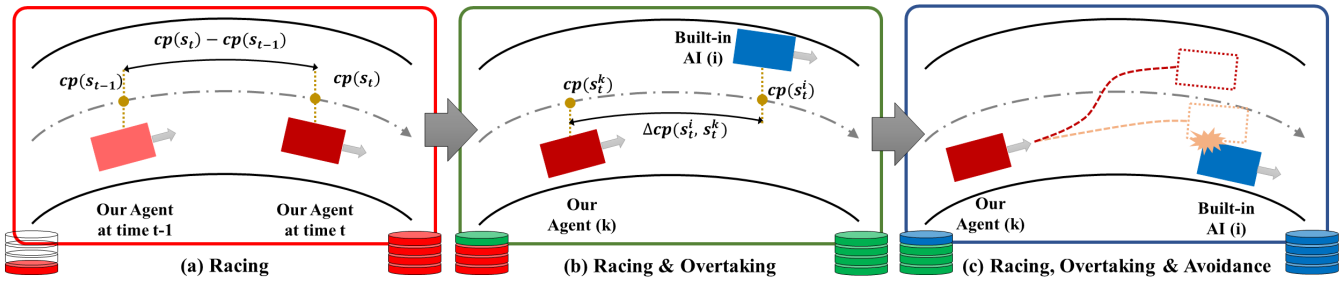


Fig. 2: An illustration of the proposed three-stage curriculum reinforcement learning for autonomous race car overtaking.

hyperparameter that trades off between the aggressiveness of the overtaking maneuver and the collision penalty. Here, $\rho_c$ is a binary flag indicating whether or not a car collision occurs, and $c_c \geq 0$ weights the car collision penalty.

The idea behind using the subtraction $(\Delta cp(s_{t-1}^i, s_{t-1}^k) - \Delta cp(s_t^i, s_t^k))$ inside the summation symbol in Eq. (2) is to continuously encourage our agent $(k)$ to approach the front opponent vehicle $(i)$ when it is driving behind, while keep maximizing the relative distance once it has overtaken the opponent vehicle $(i)$. The illustration of the proposed idea is depicted in Fig. 2. By maximizing the proposed overtaking reward, our agent can learn to perform overtaking maneuvers and collision avoidance.

### B. Observation and Action

The definition of both the observation space and the action space is described in Table I. We denote the observation vector as $\boldsymbol{o}_t = [\boldsymbol{v}_t, \dot{\boldsymbol{v}}_t, \theta, \boldsymbol{d}_t, \delta_{t-1}, f_t, f_c, \boldsymbol{c}_L]$, and use them as the input to our neural networks. The input normalization is important in most learning algorithms since different features might have totally different scales. We apply $z$-score normalization to all features in the observation vector, except for the 2D Lidar measurements $\boldsymbol{d}_t$, which is normalized using min-max normalization. We compute the $z$-score normalization using sampled states from the environment. The control actions are the steering angle and the combined throttle and brake signal, denoted as $[\delta_t, \omega_t]$ separately.

The 2D Lidar distance measurements detect the relative distance between our agent and other objects, such as other vehicles and the wall. We use a distance vector $\mathbf{d}_t \in \mathbb{R}^{72}_{>0}$ obtained from a set of 72 equally spaced Lidar beams with a maximum detection range of 20 m arranged between $-108^\circ \sim 108^\circ$ in front of vehicle. We constrain both the field-of-view and the detection range for a fair comparison to the human driver.

### C. Curriculum Soft Actor-Critic

We use Soft Actor-Critic (SAC) for training a neural network policy that can maximize the overtaking reward. However, like most off-policy algorithms, SAC suffers from "extrapolation error", a phenomenon in which unseen state-action pairs are erroneously estimated to have unrealistic values [20]. For example, in the overtaking task, it might take the agent many explorations in order to see a single overtaking since it does know how to drive at the early training stage. Hence, maximizing the overtaking reward directly leads to premature convergence and results in poor final policy.

TABLE I: The observation space and the action space.

| | | |
|---|---|---|
| Observation Space ($\mathbb{R}^{96}$) | | |
| $\boldsymbol{v}_t$ | linear velocity in body frame | $\mathbb{R}^3$ |
| $\dot{\boldsymbol{v}}_t$ | linear acceleration in body frame | $\mathbb{R}^3$ |
| $\theta$ | angle between heading and tangent to the centerline | $\mathbb{R}$ |
| $\boldsymbol{d}_t$ | 2D Lidar measurements (-108° ∼108°, 20m) | $\mathbb{R}^{72}_{>0}$ |
| $\delta_{t-1}$ | previous steering command | $\mathbb{R}$ |
| $f_t$ | binary flag with 1 indicating wall collision | $\mathbb{R}$ |
| $f_c$ | binary flag with 1 indicating car collision | $\mathbb{R}$ |
| $\boldsymbol{c}_L$ | looking forward curvature of centerline (0.2∼3.0 sec) | $\mathbb{R}^{14}$ |
| Action Space ($\mathbb{R}^2$) | | |
| $\delta_t$ | steering angle in rad, $\delta_t \in [-\pi/6, \pi/6]$ | $\mathbb{R}$ |
| $w_t$ | combination of throttle ($w_t > 0$) and brake ($w_t < 0$) | $\mathbb{R}$ |

*1) Three-stage Curriculum Reinforcement Learning:* A key ingredient to address this problem for a complex environment is to use curriculum learning. In particular, we combine SAC with a 3-stage curriculum learning procedure. In stage one, we train a policy (with random weights) for high-speed racing. We use a randomly initialized neural network and train it for the single-player racing (without overtaking) by maximizing the racing reward function (Eq. (1)). We stop the training when the agent is capable of driving the car at a very high speed. In stage two, we continuously train the same policy for aggressive racing and overtaking. We load the pre-trained policy done in the first stage and reconfigure the racing environment by adding an extra vehicle, which is controlled by the built-in game AI controller. We initialize the distance between our agent and the build-in agent with 200 meters separation along the centerline. Before training, it is important to keep the old replay buffer, and reinitialize the weights of the exploration term in the stochastic policy in that the policy maintains sufficient explorations. We update the policy by maximizing the overtaking reward (Eq. 2) and using new sampled trajectories. In stage three, we obtain a final policy that can race the car at high speed, overtake its opponents, and avoid collisions. This is achieved by

increasing the penalty term in the overtaking reward and training the policy with new samples. It is important to use a fixed size first-in-first-out replay buffer, since the racing data will be replaced gradually with new overtaking samples.

*2) Distributed Sampling Strategy:* The second key ingredient in achieving better global convergence for complex environments is to use a distributed sampling strategy for the data collection. Similar to [6], we use a distributed sampling strategy, in which we use multiple simulators (4) in parallel, each simulating multiple cars (20) on the same racing track. In other words, we can achieve $4 \times 20$ faster sampling speed than using a single racing environment. Most importantly, the sampled trajectories cover most of the track segments, and led to a dataset that highly correlates with the true state-action distribution. As a result, we achieve fast data collection and stable policy training. We use this sampling strategy throughout all training stages.

## IV. EXPERIMENTS

We design experiments to answer the following research questions:

- Can our curriculum learning speed up training and improve sample efficiency in comparison with standard training (Section IV-B)?
- How should we evaluate the overtaking performance (Section IV-C)?
- What are the overtaking strategies learned by our approach? (Section IV-D)?

### A. Experimental Setup

We conduct our experiment using Gran Turismo Sport (GTS). We train our algorithm on a desktop with an i7-8700 CPU and a GTX 1080Ti GPU. We use a custom implementation [6] of the Soft Actor-Critic algorithm that is based on the open-source baselines [21]. GTS runs on a PlayStation 4, and we interact with GTS using an Ethernet connection. We treat GTS as a black-box simulator since we do not have direct access to the vehicle dynamics and the environment in GTS. We choose "Audi TT Cup 16" as the simulated car model and "Tokyo Expressway - Central Outer Loop" as the race track. The hyperparameters of SAC for the training are listed in TABLE II.

TABLE II: Hyperparameters

| Hyperparameter | Value |
|---|---|
| Neural network structure (MLP) | $2 \times$ [256, ReLU] |
| Mini-batch size | 4,096 |
| Replay buffer size | $1 \times 10^6$ |
| Start step (a trick to improve exploration) | $4 \times 10^4$ |
| Learning rate | 0.001 |
| Exponential discount factor | 0.99 |
| Episode steps (under 10 Hz sampling rate) | 1,000 |
| Total steps per epoch (20 cars in parallel) | 20,000 |

### B. Curriculum Policy Training for Overtaking

The first step towards autonomous overtaking in car racing is to obtain a policy that can drive faster than the opponents (the built-in AI) in a single-car racing environment. The built-AI uses a rule-based approach to follow a predefined trajectory, similar to [22], [23]. We study different approaches to obtain such a policy before learning to overtake, including naive Behavior Cloning (BC) [24], Generative Adversarial Imitation Learning (GAIL) [25], Deep Planning Network (PlaNet) [26], and Twin Delayed Deep Deterministic Policy Gradient (TD3) [27]. We use the same neural network structure and observation representation for all methods. The training data required by imitation learning is collected using demonstrations from both the built-AI and human experts. The experimental result is shown in TABLE III, only the policy trained using the model-free RL (SAC and TD3) can outperform the built-in AI in the single-car time trial race.

TABLE III: A baseline comparison for the single-car race.

| | Built-in AI | BC | GAIL | SAC | TD3 | PlaNet |
|---|---|---|---|---|---|---|
| Lap Time (s) | 86.9 | 108.0 | 144.9 | 80.1 | 80.8 | 89.0 |
| Average Speed ($\mathrm{km\,h^{-1}}$) | 184.4 | 146.8 | 109.4 | 198.2 | 196.3 | 178.2 |

To understand the effect of the proposed curriculum learning on policy training, we compare the training curves of curriculum SAC with standard SAC. For standard SAC, we train neural network policies by directly maximizing the overtaking reward (Eq. (2)). By contrast, for curriculum SAC, we first design a single-player racing environment and train a neural network policy by maximizing the racing reward (Eq. (1)). Then, we configure an overtaking environment and continuously train the policy by maximizing the overtaking reward (Eq. (2)) with collision weights of $c_w = c_c = 0.005$. The learning curves are shown in Fig. 3. As a result, the proposed curriculum SAC outperforms standard SAC in terms of sample efficiency and final policy performance.

### C. Evaluation of the Overtaking Performance

Evaluating the overtaking performance can be complicated as there are multiple metrics, such as the total travel time or the total collision time. These values are generally difficult to obtain in the real world. GTS provides precise quantitative measurements of those metrics. We propose four objective evaluation metrics for evaluating the obtained policy: 1) total travel time, 2) total travel distance, 3) total car collision time, and 4) total wall collision time.

We train three different agents using the overtaking reward with different hyperparameters and different training procedures for benchmark comparisons. For Agent1, we use only the first 2 stages that include single-car racing and multiple-car overtaking. For Agent2, we use 3-stage training,

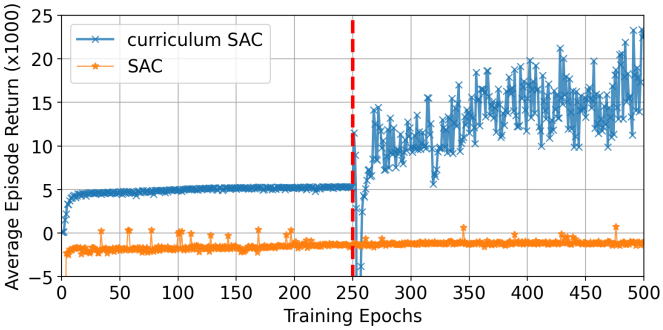


Fig. 3: A comparison of the learning curves using different training methods. The red dash line in the middle represents the switch from stage one to stage two.

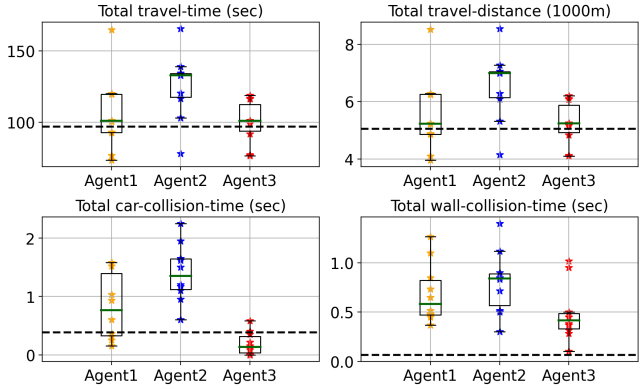


Fig. 4: Evaluation comparisons for the setting A. The dashed lines indicate the human player's best performance.

where the second and the third stage has same collision weights of $c_w = c_c = 0.005$. For Agent3, we use 3-stage training, where the third stage has larger collision weights of $c_w = c_c = 0.01$ than the second stage, which has collision weights of $c_w = c_c = 0.005$.

To evaluate the overtaking performance of 3 trained agents, we use two different settings for the evaluation experiment. We place 5 opponent vehicles in front of the trained agent with an initial separation distance of $50\,\mathrm{m}$ (setting A) and of $200\,\mathrm{m}$ (setting B). In addition, we invite an expert player TG (name omitted for reasons of anonymity) as a human baseline. Both the human player and our agent use exactly the same settings and have to overtake all the 5 opponent vehicles.

We compute the evaluation metrics for each trained agent by conducting the experiment repeatedly 10 times, and for the human player by repeating the same experiment 2 times. We take the best result from the human player as our baseline. The evaluation results are shown in Fig. 4 and Fig. 5. Both our agents and the human player are capable of overtaking all 5 opponent vehicles. Our agents achieve comparable overtaking performance as the human expert in setting A.

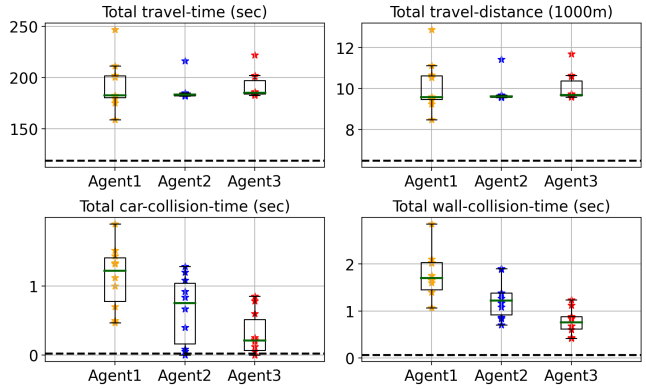


Fig. 5: Evaluation comparisons for the setting B. The dashed lines indicate the human player's best performance.

### *D. Learned Overtaking Behaviors*

To understand the overtaking strategy learned by our approach, we conduct a detailed analysis of the executed overtaking trajectory. We compare the fastest trajectory executed by our agent (Agent3) with the fastest trajectory performed by the human player, both experiments are conducted using setting A. Fig. 6 (*Top*) shows a direct comparison of the overtaking progress between our agent and the human expert. In this comparison, it takes our agent less time to overtake all 5 front cars than that of the human driver. However, our agent drives at high-speed which leads to more collisions with its opponents and wall. Overall, our agent shows a comparable overtaking performance against the human expert.

The trajectory plots in Fig. 6 (*Middle*) show five overtaking trajectories (red dashed lines) performed by our agent and the trajectories (black solid lines) executed by the built-in game AI. The speeds are colored according to the color bar on the right. Our agent can maintain high-speed driving during the overtaking. Besides, our agent demonstrates different overtaking strategies in different driving scenarios, depending on both the track segment and the opponents' driving strategy. For example, the first and the second overtaking occurred consecutively on a difficult track segment, which has a sharp turn. Our agent learns to drive along the outer side of the track at high speed since it has more free space. In addition, our agent manages to overtake its opponents on straight segments of the track. The screenshots provide a visualization of five different overtaking moments.

As a comparison, the plots on the bottom show the overtaking trajectories performed by the human player. In summary, the human player can also overtake all the front vehicles, but drives trajectories that are strategically different from those learned by our agent. For example, in the first overtaking segment, the human player took the inner side of the track, and hence, has to largely decrease its speed when entering the curve. Similarly, the human player can also perform overtaking on straight segments of the track by simply speeding up the vehicle.

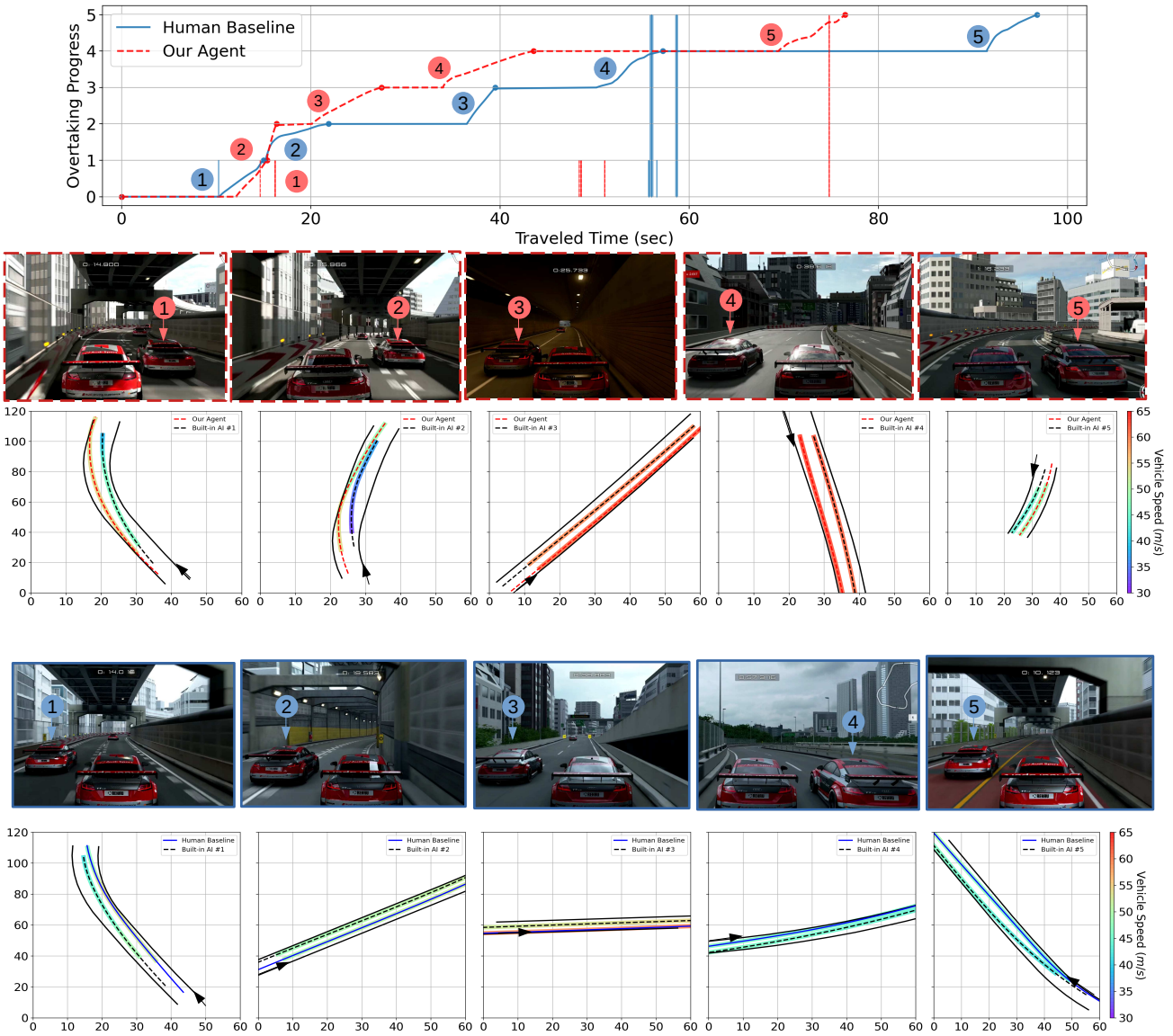


Fig. 6: *Top:* A comparison of the overtaking progress between our agent and an experienced human driver. The long vertical straight lines indicate car collisions and the short vertical lines show wall collisions. *Screenshots:* A visualization of five overtaking moments by our agent and the human player. *Trajectories:* The overtaking trajectories learned by our agent, the trajectories executed by the built-in game AI, and the overtaking trajectories performed by the human expert.

## V. CONCLUSION

In this work, we proposed the usage of curriculum reinforcement learning to tackle high-speed autonomous race-car overtaking in Gran Turismo Sport. We demonstrated the advantages of curriculum RL over standard RL in autonomous overtaking, including better sample efficiency and overtaking performance. The learned overtaking policy outperforms the built-in model-based game AI and achieves comparable performance with an experienced human driver.

Our empirical analysis suggests that complex tasks that are difficult to solve from scratch can be first sequenced into a curriculum and, then, be solved more efficiently with a stage-by-stage learning procedure. The proposed approach has limitations in terms of scalability and generalizability. In particular, the learned control policies are validated only in simulation and restricted to apply to a single track/car combination. Nevertheless, the method presented in this paper can serve as a step towards developing more practical autonomous-driving systems in the real world.

## VI. ACKNOWLEDGMENTS

We thank Kenta Kawamoto and Florian Fuchs from Sony AI Tokyo and Zurich, respectively, for their help and fruitful discussions.

## REFERENCES


[1] S. Dixit, S. Fallah, U. Montanaro, M. Dianati, A. Stevens, F. Mccullough, and A. Mouzakitis, "Trajectory planning and tracking for autonomous overtaking: State-of-the-art and future prospects," *Annual Reviews in Control*, vol. 45, pp. 76–86, 2018.

[2] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Transactions on intelligent vehicles*, vol. 1, no. 1, pp. 33–55, 2016.

[3] A. Buyval, A. Gabdulin, R. Mustafin, and I. Shimchik, "Deriving overtaking strategy from nonlinear model predictive control for a race car," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 2623–2628.

[4] A. Heilmeier, A. Wischnewski, L. Hermansdorfer, J. Betz, M. Lienkamp, and B. Lohmann, "Minimum curvature trajectory planning and control for an autonomous race car," *Vehicle System Dynamics*, pp. 1–31, 2019.

[5] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science robotics*, vol. 5, no. 47, 2020.

[6] F. Fuchs, Y. Song, E. Kaufmann, D. Scaramuzza, and P. Duerr, "Super-human performance in gran turismo sport using deep reinforcement learning," *arXiv preprint arXiv:2008.07971*, 2020.

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

[8] L. Ma, J. Xue, K. Kawabata, J. Zhu, C. Ma, and N. Zheng, "A fast rrt algorithm for motion planning of autonomous road vehicles," in *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2014, pp. 1033–1038.

[9] Y. Kuwata, G. A. Fiore, J. Teo, E. Frazzoli, and J. P. How, "Motion planning for urban driving using rrt," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2008, pp. 1681–1686.

[10] S. Dixit, U. Montanaro, M. Dianati, D. Oxtoby, T. Mizutani, A. Mouzakitis, and S. Fallah, "Trajectory planning for autonomous high-speed overtaking in structured environments using robust mpc," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 6, pp. 2310–2323, 2019.

[11] M. Wang, Z. Wang, J. Talbot, J. C. Gerdes, and M. Schwager, "Game theoretic planning for self-driving cars in competitive scenarios," in *Robotics: Science and Systems*, 2019.

[12] P. Petrov and F. Nashashibi, "Modeling and nonlinear adaptive control for autonomous vehicle overtaking," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 4, pp. 1643–1656, 2014.

[13] W. Farag and Z. Saleh, "Behavior cloning for autonomous driving using convolutional neural networks," in *2018 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT)*. IEEE, 2018, pp. 1–7.

[14] D. A. Pomerleau, "Efficient training of artificial neural networks for autonomous navigation," *Neural computation*, vol. 3, no. 1, pp. 88–97, 1991.

[15] W. Schwarting, T. Seyde, I. Gilitschenski, L. Liebenwein, R. Sander, S. Karaman, and D. Rus, "Deep latent competition: Learning to race using visual control policies in latent space," in *Conference on Robot Learning*, 2020.

[16] X. Li, X. Xu, and L. Zuo, "Reinforcement learning based overtaking decision-making for highway autonomous driving," in *2015 Sixth International Conference on Intelligent Control and Information Processing (ICICIP)*, 2015, pp. 336–342.

[17] D. Loiacono, A. Prete, P. L. Lanzi, and L. Cardamone, "Learning to overtake in torcs using simple reinforcement learning," in *IEEE Congress on Evolutionary Computation*. IEEE, 2010, pp. 1–8.

[18] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 2011, pp. 627–635.

[19] R. Verschueren, S. De Bruyne, M. Zanon, J. V. Frasch, and M. Diehl, "Towards time-optimal race car driving using nonlinear mpc in real-time," in *53rd IEEE conference on decision and control*. IEEE, 2014, pp. 2505–2510.

[20] S. Fujimoto, D. Meger, and D. Precup, "Off-policy deep reinforcement learning without exploration," in *International Conference on Machine Learning*, 2019, pp. 2052–2062.

[21] J. Achiam, "Spinning Up in Deep Reinforcement Learning," 2018.

[22] J. Ni, J. Hu, and C. Xiang, "Robust path following control at driving/handling limits of an autonomous electric racecar," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 6, pp. 5518–5526, 2019.

[23] T. Hellstrom and O. Ringdahl, "Follow the past: a path-tracking algorithm for autonomous vehicles," *International journal of vehicle autonomous systems*, vol. 4, no. 2-4, pp. 216–224, 2006.

[24] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," in *Advances in neural information processing systems*, 1989, pp. 305–313.

[25] J. Ho and S. Ermon, "Generative adversarial imitation learning," in *Advances in neural information processing systems*, 2016, pp. 4565–4573.

[26] D. Hafner, T. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, and J. Davidson, "Learning latent dynamics for planning from pixels," in *International Conference on Machine Learning*. PMLR, 2019, pp. 2555–2565.

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