Max-Q Verstappen: Performing F1 Hot Laps with RL

Abhay Deshpande, Arnav Thareja

December 12, 2022

1 Introduction

With recent developments in the field of autonomous vehicles, the problem of autonomous racing has seen a rise in popularity among researchers. The problem focuses on autonomous driving scenarios in structured, high-speed domains that are designed to push vehicles to their limits. Advances in autonomous racing can be applied to the general problem of autonomous driving or advanced driver assistance systems (ADAS), including improved and lower latency control of vehicles, safer navigation at higher speeds, and safer interactions between vehicles. A number of competitions and testing environments have been established to help further developments in this field, including F1TENTH[1], Formula Student Driverless[2], Indy Autonomous Challenge[3], and Roborace[4]. These competitions focus on both minimizing single-agent lap times and winning multi-agent races against other teams.

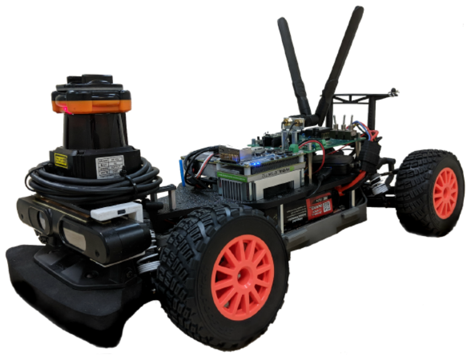

Figure 1: An example of an F1-Tenth car

[1]https://f1tenth.org/

[2]https://www.fsaeonline.com/

[3]https://www.indyautonomouschallenge.com/

[4]https://roborace.com/

Our final project uses deep reinforcement learning to perform hot laps[5] in a simulator with F1-style cars. We use the F1TENTH[6] [6] gym environment, which provides a simulation of 1/10th scale F1 cars with accurate dynamics modeling. We train a model using the SAC [5] algorithm on the F1 Circuit of the Americas (COTA) track, and show transfer of our trained model onto other F1 tracks previously unseen by the model. Our model performs comparably to human lap records, and achieves superhuman performance on some tracks.

2 Related Work

Multiple approaches exist to tackle autonomous racing in an F1 environment, some of these using the same F1TENTH environment. In TC-Driver [4], the authors train an agent to track a trajectory on an F1 circuit. The RL-based model shows a lower crash rate than an MPC, since it can better handle tire forces that the MPC does not consider or models inaccurately. The trained model is outperformed by MPC in speed and lap times, since it is evaluated while tracking the centerline of the track, which is a suboptimal trajectory that gives slower lap times.

In [10], the authors train an agent to complete laps as fast as possible in the F1TENTH environment. The authors start with a modified artificial potential field (MAPF) controller and use reinforcement learning to obtain a policy that learns residual actions to improve the policy from MAPF. The authors train their model using both the TRPO [8] and PPO [7] algorithms, and the trained model is robust to varying tire friction.

In [3], the authors train an agent in the Gran Turismo Sport simulator that outperforms the best human drivers on the same tracks. Their agent minimizes the total lap time while avoiding crashing into barriers, and they train this agent using the SAC [5] algorithm. The trained agent is robust to modifications to the track environment, including different tracks, different car dynamics, and observation noise.

Our approach differs from these related works in that instead of training an RL model to augment a previous controller, like the MPC as in TC-Driver [4] or the MAPF as in ResRace [10], we train an end-to-end model for racing that is not dependent on other existing controllers. In contrast to [3], we focus on a different environment, with different rules and a different reward function.

3 Approach

3.1 State Representation

We trained the agent using a predesigned featurization of the state, using similar features as described in [3]. This featurization is represented as a 37-dimensional vector, detailing the following information:

[5]A hot lap is a lap performed as fast as possible, without considering other cars or other factors in racing
[6]`https://github.com/f1tenth/f1tenth_gym`

- 2D velocity in local reference frame
- 2D acceleration in local reference frame
- Angle from centerline (The signed angle difference between the agent's current heading and the heading of the centerline)
- Future curvatures of track (10 signed curvature samples taken from 0.5m increments in front of the agent along the centerline)
- 20 LIDAR samples taken uniformly from a 270° span centered at the front of the agent
- The previous steer angle
- A boolean indicating whether the agent is in a collision or not

Notably, we designed this featurization function to be completely local to the pose of the agent. This allows it to generalize optimal behavior between different parts of the track, and even to entirely new tracks.

3.2 Reward Function

We formulated our reward function as

$$r = v_x - k_{cen} v_x^2 |\tan(\delta)| - k_{skid}|v_y| - \begin{cases} k_{coll}\sqrt{v_x^2 + v_y^2} & \text{if in collision} \\ 0 & \text{otherwise} \end{cases}$$

where

$$\begin{aligned} v_x &= \text{Velocity in the forward direction} \\ v_y &= \text{Velocity in the left direction} \\ \delta &= \text{Current steer angle} \\ k_{cen} &= \text{Centripetal acceleration penalty weight} \\ k_{skid} &= \text{Skidding penalty weight} \\ k_{coll} &= \text{Collision penalty weight} \end{aligned}$$

Our trained model uses $k_{cen} = 0.05$, $k_{skid} = 0.5$, and $k_{coll} = 5$.

Note that using single track kinematics, $R = L/\tan(\delta)$, so $v_x^2 \tan(\delta) \propto v_x^2/R$, making it a suitable proportional approximation of centripetal acceleration.

From this formulation, we see that the reward function incentivizes higher forward velocity while penalizing centripetal acceleration, skidding, and collision. High centripetal acceleration is the precursor to losing traction while skidding represents the actual loss of traction, so it makes sense to penalize both of these phenomena. Penalizing collision is important too, since while all collisions terminate the episode, weighting the collision based on the vehicle speed breaks up the sparsity of this signal, indicating to the agent that high impact collisions can be avoided by going slower.

3.3 Training the Agent

We used model-free deep reinforcement learning to solve this challenge, due to its ability to tackle a wide variety of difficult challenges, as well as not requiring a system model to operate, making it more flexible and robust. Specifically, we used the SAC [5] algorithm implementation in the D3RLPY [9] library. For our experiments, we trained the model exclusively on COTA, and later note that it is able to transfer to new tracks successfully.

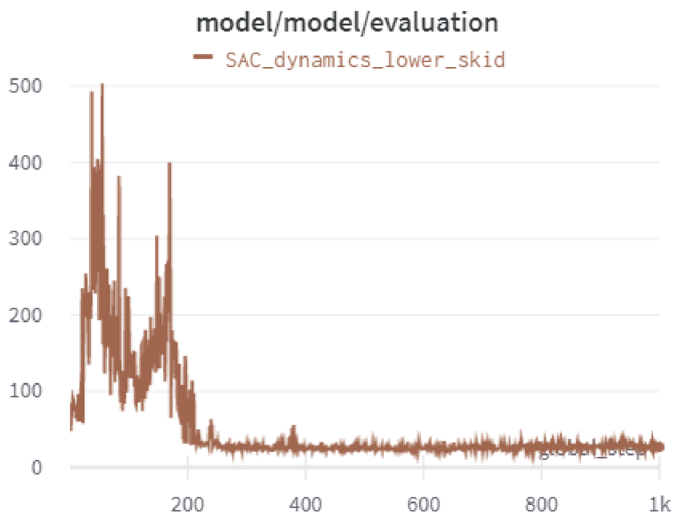

Figure 2: An evaluation curve of an agent that experiences catastrophic forgetting. Note that around epoch 200 the evaluation score plummets to about 30 and never improves.

While training models, we ran into a few difficulties. First, we originally tried a much simpler reward function that only rewarded velocity, reasoning that the critic in SAC would learn to avoid skidding and collisions. However, we found that this was much worse at training, since the agent could only learn that skidding was bad through colliding and terminating the episode, leading to a sparse penalty signal that became harder to learn. We solved this problem by augmenting the reward function, adding the centripetal acceleration, skidding, and collision penalties as described above. Additionally, another problem we found was catastrophic forgetting, where the agent "forgets" the basics on how to operate, and performance suddenly plummets, as shown in the train graph in figure 2. We countered this by increasing the size of the replay buffer used by the agent, which allowed it to sample past transitions and retain that knowledge, avoiding the forgetting problem.

4 Results

By employing the approach described above, we successfully trained an agent to compete with F1 drivers, and even outperform them on some tracks. Figure 3 shows our agent's performance compared to the historic records from human drivers in F1, on multiple real-life tracks taken from [1]. Note that our model was trained was trained only on COTA, but still achieves superhuman performance in Mexico City, while remaining somewhat competitive on the other tracks.

Figure 4 illustrates the trajectory our agent takes around COTA and Interlagos, compared to the optimal raceline. From these visualizations, it's apparent that our agent takes a conservative estimate of the raceline, erring on the side of caution and staying more towards the center. From the agent's perspective this makes sense, since it minimizes any risk of losing traction and going into the wall. Better performance could potentially be achieved by reformulating the reward function to make the agent drive more aggressively to post faster laptimes.

Track	RL Agent (ours)	F1 Lap Record
COTA	**90.00s**	92.03s
Interlagos	69.40s	**67.28s**
Silverstone	95.95s	**84.30s**
Mexico City	**71.10s**	74.76s
Red Bull Ring	69.25s	**62.94s**

Figure 3: Our agent's best lap time over 10 laps compared to the current F1 record[7]

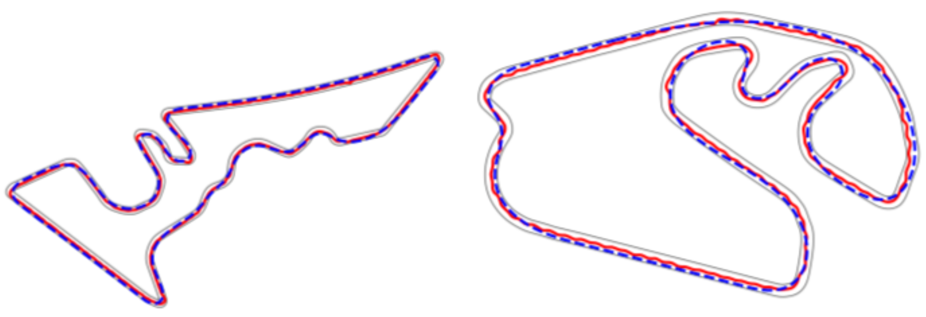

Figure 4: Our agent's path around COTA and Interlagos (red) compared to the mathematically optimal trajectory (blue)

5 Discussion

After training our model, we noticed that it exhibited suboptimal behavior in some scenarios, for example, weaving on straights, where driving straight would yield a faster lap time. We suspect this stems from our formulation of the reward function, and would experiment with varying penalty weights to address this.

[7]According to https://fastestlaps.com/

Our trained model may be outperformed by MPC or similar planning algorithms, which are able to access global state and further lookahead, in contrast to the local state features we use. To mitigate this, we can investigate using MPC or planning to inform the training of the policy such as in [2], or learning a residual policy as in [10].

Future work on this project could focus on extending results to the multiagent domain. Many of the existing autonomous racing competitions, including F1TENTH, have a head-to-head racing competition, where 2 or more agents compete against each other for the fastest race times. This adds additional complexity in predicting the motion of the other agents, and using these predictions to overtake and avoid collisions. Further work in this direction would be applicable to real-life driving scenarios, since many cars share the road in real life.

Another potential direction to explore would involve transferring our learned model onto real-world cars. This would introduce challenges in extracting our state featurization from sensor data instead of a provided simulation environment, and dealing with any simulation inaccuracies in dynamics modeling.

6 Citations

[1] Johannes Betz et al. "Autonomous vehicles on the edge: A survey on autonomous vehicle racing". In: *IEEE Open J. Intell. Transp. Syst.* 3 (2022), pp. 458–488. DOI: 10.1109/ojits.2022.3181510.

[2] Mohak Bhardwaj et al. *Information Theoretic Model Predictive Q-Learning*. 2020. DOI: 10.48550/ARXIV.2001.02153. URL: https://arxiv.org/abs/2001.02153.

[3] Florian Fuchs et al. "Super-Human Performance in Gran Turismo Sport Using Deep Reinforcement Learning". In: *IEEE Robotics and Automation Letters* 6.3 (July 2021), pp. 4257–4264. DOI: 10.1109/lra.2021.3064284. URL: https://doi.org/10.1109%2Flra.2021.3064284.

[4] Edoardo Ghignone et al. *TC-Driver: Trajectory Conditioned Driving for Robust Autonomous Racing – A Reinforcement Learning Approach*. 2022. DOI: 10.48550/ARXIV.2205.09370. URL: https://arxiv.org/abs/2205.09370.

[5] Tuomas Haarnoja et al. *Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor*. 2018. DOI: 10.48550/ARXIV.1801.01290. URL: https://arxiv.org/abs/1801.01290.

[6] Matthew O'Kelly et al. "F1TENTH: An Open-source Evaluation Environment for Continuous Control and Reinforcement Learning". In: *NeurIPS 2019 Competition and Demonstration Track*. PMLR. 2020, pp. 77–89.

[7] John Schulman et al. *Proximal Policy Optimization Algorithms*. 2017. DOI: 10.48550/ARXIV.1707.06347. URL: https://arxiv.org/abs/1707.06347.

[8] John Schulman et al. *Trust Region Policy Optimization*. 2015. DOI: 10.48550/ARXIV.1502.05477. URL: https://arxiv.org/abs/1502.05477.

[9] Takuma Seno and Michita Imai. “d3rlpy: An Offline Deep Reinforcement Learning Library”. In: *Journal of Machine Learning Research* 23.315 (2022), pp. 1–20. URL: http://jmlr.org/papers/v23/22-0017.html.

[10] Ruiqi Zhang et al. “Residual Policy Learning Facilitates Efficient Model-Free Autonomous Racing”. In: *IEEE Robotics and Automation Letters* 7.4 (2022), pp. 11625–11632. DOI: 10.1109/LRA.2022.3192770.