– Independent Work Report Spring 2025 –

Safe Reinforcement Learning for Autonomous Racing: Training a Safety Filter From Scratch

Eshaan Govil
Advisor(s): Prof. Jaime Fernández Fisac, Donggeon David Oh

Abstract

Safe reinforcement learning (RL) is critical for deploying autonomous agents in real-world, safety-critical domains such as high-speed racing. However, traditional RL methods prioritize reward maximization without explicit constraint enforcement, making them unsuitable for environments where failure must be avoided. This work presents a proof-of-concept implementation of a least-restrictive safety filter (LRSF) for reinforcement learning, inspired by Hamilton–Jacobi (HJ) reachability theory. A learned safety critic is integrated into the Soft Actor-Critic (SAC) algorithm, enabling runtime intervention only when proposed actions are predicted to violate safety constraints. Using the classic `CartPole-v1` *environment as a simplified racing proxy, it is demonstrated that safety-aware agents can not only maintain constraints but also achieve improved performance by avoiding premature failures. The results suggest that learned safety filters can act as lightweight stabilizers during training and effective protectors during deployment. This project lays the groundwork for scaling safe RL methods to complex autonomous racing tasks, with the ultimate goal of deploying these architectures on physical platforms such as the PAVE autonomous go-kart.*

1. Introduction

Reinforcement learning (RL) has shown tremendous potential in solving complex control problems, from Atari games [5] to robotic manipulation. However, its application in safety-critical domains, such as autonomous driving and high-speed racing, remains limited due to the lack of safety guarantees. Agents trained purely to maximize reward often exploit unsafe behaviors, especially early in training, when exploration dominates [10]. This makes traditional RL unsuitable for real-world systems operating under tight safety constraints.

Autonomous racing is a particularly compelling yet challenging domain for safe RL. Racing agents must operate near the limits of physical feasibility, balancing aggressive maneuvers with collision avoidance, track adherence, and long-term viability. Although model-based techniques such as control barrier functions (CBFs) or Hamilton–Jacobi (HJ) reachability analysis offer formal safety guarantees, they often require known system dynamics and do not scale well to high-dimensional or black-box environments.

To address this, recent work has explored learning-based approximations of safety value functions—using ideas from HJ theory—to synthesize *safety filters* that override unsafe actions at runtime. These filters intervene only when necessary, allowing agents to prioritize constraint satisfaction without fully sacrificing reward.

In this work, such a filter will be implemented and evaluated in a proof-of-concept setting: the classic `CartPole-v1` environment. Although simple, CartPole provides a well-structured, interpretable platform to test safety-critical behavior, with pole angle and cart position acting as stand-ins for system constraints.

This paper lays the introductory work for transferring these insights back into the autonomous racing domain, with the long-term goal of deploying safe RL policies on high-speed simulators and physical platforms. This project is part of a larger effort founded by Princeton Autonomous Vehicle Engineering (PAVE), a "student-led research group at Princeton University focused on the advancement of robotics and artificial intelligence through a variety of research and competition

with land and maritime vehicles" [8]. The overall goal of PAVE's project, titled *Ivy: Racer*, is to develop a budget-friendly autonomous go-kart racing league.

2. From Normal Reinforcement Learning to Safety-Conscious Control

In standard reinforcement learning, the objective is to find a policy $\pi(a|s)$ that maximizes the expected discounted return:

$$V^{\pi}(s) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \,\middle|\, s_0 = s\right],$$

where $\gamma \in [0,1)$ is the discount factor ensuring convergence. The associated optimal value function $V^*(s)$ satisfies the classical Bellman optimality equation:

$$V^*(s) = \max_{a \in \mathbb{A}} \left[r(s,a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)}[V^*(s')]\right].$$

This recursive structure underpins dynamic programming algorithms such as value iteration and Q-learning.

However, in safety-critical domains, the aim is not to maximize accumulated reward, but rather to *avoid failure*. Formally, we define a failure set $\mathbb{F} \subset \mathbb{S}$ and a safety margin function $g : \mathbb{S} \to \mathbb{R}$, where $g(s) < 0$ if and only if $s \in \mathbb{F}$. The safety objective becomes remaining within the safe set $\mathbb{F}^c$ indefinitely.

This motivates defining the safety value function:

$$V(s) = \sup_{\pi} \inf_{t \geq 0} g(s_t),$$

which represents the best worst-case safety margin achievable starting from s. Unlike cumulative rewards, this minimum-over-time structure breaks additivity, and the corresponding dynamic programming operator is *not* a contraction. As a result, iterative methods like value iteration may fail to converge.

2.1. Discounted Safety Value Functions

To address this, Fisac et al. [3] proposed a time-discounted safety formulation, modifying the value recursion to:

$$(T_\gamma V)(s) = (1-\gamma)g(s) + \gamma \max_{a\in\mathbb{A}} V(f(s,a)),$$

where $f(s,a)$ denotes the system's next state under action a.

Contraction Property. We now prove that T_γ is a γ-contraction with respect to the sup norm $\|\cdot\|_\infty$.

Let $V, W : \mathscr{S} \to \mathbb{R}$ be two bounded functions. Then:

$$\begin{aligned}
|(T_\gamma V)(s) - (T_\gamma W)(s)| &= \gamma \left| \max_a V(f(s,a)) - \max_a W(f(s,a)) \right| \\
&\leq \gamma \sup_a |V(f(s,a)) - W(f(s,a))| \\
&\leq \gamma \|V - W\|_\infty.
\end{aligned}$$

Taking the supremum over s, we conclude:

$$\|T_\gamma V - T_\gamma W\|_\infty \leq \gamma \|V - W\|_\infty,$$

which establishes contraction.

Thus, by Banach's Fixed Point Theorem, T_γ has a unique fixed point V_γ^*, and value iteration converges to V_γ^* from any initial guess.

Limit Behavior. As $\gamma \to 1$, the fixed point V_γ^* approaches the true minimum-over-time safety value function:

$$\lim_{\gamma\to 1} V_\gamma^*(s) = \sup_\pi \inf_{t\geq 0} g(s_t).$$

Thus, by solving the discounted problem with large γ, we approximate the original undiscounted safety objective arbitrarily closely.

This formulation enables scalable, learning-based safety analysis using Q-learning or actor-critic methods, bridging reinforcement learning and control-theoretic Hamilton–Jacobi reachability.

3. Translating from Vanilla SAC to Safety SAC

Soft Actor-Critic (SAC) [4] is a powerful off-policy RL algorithm that augments the expected return with an entropy regularization term:

$$J(\pi) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t \left(r(s_t, a_t) + \alpha \mathbb{H}(\pi(\cdot|s_t))\right)\right],$$

where $\mathbb{H}(\pi(\cdot|s)) = -\mathbb{E}_{a\sim\pi(\cdot|s)} \log \pi(a|s)$ is the entropy and $\alpha > 0$ is the temperature.

This formulation encourages both reward maximization and policy stochasticity, leading to better exploration and robustness.

However, vanilla SAC lacks any explicit mechanism to enforce safety constraints. Agents can learn unsafe high-reward policies that violate critical constraints—unacceptable in domains like autonomous driving or racing.

3.1. Safety Critic and Discounted Safety Bellman Update

To endow SAC with safety-awareness, we introduce a **safety critic** $Q^{\text{safe}}(s,a)$, estimating future safety margins.

The target for Q^{safe} is based on the discounted safety Bellman operator:

$$y_{\text{safe}} = (1-\gamma)g(s) + \gamma \min\left\{g(s), \max_{a'} Q^{\text{safe}}(s',a')\right\}.$$

The safety critic is trained by minimizing the TD-error:

$$\mathbb{L}_{\text{safe}} = \mathbb{E}_{(s,a,s')}\left[\left(Q^{\text{safe}}(s,a) - y_{\text{safe}}\right)^2\right].$$

Convergence Guarantee. Because the target operator is a γ-contraction (as proven above), temporal-difference learning converges to the unique fixed point Q^{safe}_{γ} under standard conditions

(boundedness, proper sampling).

3.2. Safety Filtering of Actions

During policy execution, candidate actions sampled from the learned SAC policy $\pi(\cdot|s)$ are filtered as follows:

- If $Q^{\text{safe}}(s,a) > 0$, the action is safe and executed.
- Otherwise, a fallback action $\arg\max_{a'} Q^{\text{safe}}(s,a')$ is selected.

This ensures that the agent never takes an action predicted to violate safety constraints.

3.3. Actor Update with Safety Constraints

There are two main strategies for actor training:

- **Hard Filtering**: Sample actions from $\pi(\cdot|s)$ but only execute actions satisfying $Q^{\text{safe}}(s,a) > 0$.
- **Soft Constraint Penalization**: Add a penalty term to the SAC objective:

$$J_{\text{safe}}(\pi) = J(\pi) - \beta \mathbb{E}_{\pi}\left[\mathbf{1}\{Q^{\text{safe}}(s,a) < 0\}\right],$$

where β is a penalty coefficient.

In our setup, we implement hard filtering at runtime: SAC proposes actions freely, but a learned **neural safety filter** overrides any unsafe actions, maintaining a least-restrictive intervention policy.

3.4. Key Insight: Unified Learning of Reward and Safety

By embedding the discounted safety Bellman structure within SAC, we unify reward maximization and safety constraint enforcement inside a fully differentiable, model-free architecture.

This Safety SAC framework allows agents to achieve high performance while satisfying formal safety guarantees, providing a practical foundation for domains such as high-speed autonomous racing.

4. Autonomous Racing as a Motivating Use Case

While much of reinforcement learning research focuses on games and robotic manipulation, autonomous racing offers a uniquely challenging and safety-critical domain. In racing, agents must operate near the physical limits of traction, reaction time, and vehicle stability. Success hinges not only on maximizing speed but also on avoiding catastrophic failures—such as going off-track, colliding with other vehicles, or destabilizing the vehicle itself.

These characteristics make racing an ideal proving ground for *safety-aware reinforcement learning*. Unlike navigation or cruising tasks, where conservative policies can still succeed, racing requires agents to push the boundary between aggressive behavior and safe constraint satisfaction. A purely reward-driven agent may learn to exploit unsafe maneuvers—such as cutting corners too tightly or failing to brake in time—if such behaviors result in higher short-term reward. In the absence of an explicit mechanism for enforcing safety constraints, these agents tend to prioritize performance at the cost of robustness and reliability.

Traditional model-based approaches to safe racing include Model Predictive Control (MPC), Control Barrier Functions (CBFs), and Hamilton–Jacobi (HJ) reachability analysis. While powerful, these methods typically assume known system dynamics and are computationally expensive to evaluate in real time. Moreover, they are difficult to scale to high-dimensional, nonlinear environments like those found in realistic racing simulators.

Recent work addresses these limitations by integrating HJ-inspired safety concepts into model-free reinforcement learning. In particular, the Human-Centered Safety Filter (HCSF) framework proposed by Oh et al. [6] combines a learned safety value function with a runtime intervention mechanism to enforce constraints in black-box systems. This architecture enables safe behavior in complex environments such as the Assetto Corsa racing simulator (as seen in Figure 1), without relying on hand-tuned dynamics models.

This work draws direct inspiration from these efforts. Although this implementation takes place in the low-dimensional CartPole environment [7], it can be framed as a step toward safe racing

Figure 1: Simulator set-up used in Oh et al. study

autonomy. The CartPole domain serves as a minimalist stand-in for racing: maintaining the pole upright is analogous to maintaining vehicle stability, while the cart's lateral position mimics staying on track. Failing to keep the pole within a safe range results in episode termination—just as going off track would in a race.

By prototyping a Least-Restrictive Safety Filter (LRSF) in this environment, it can be evaluated whether or not safety constraints derived from a learned Q-function can improve long-term performance while avoiding failures. This allows testing of the safety filter architecture in a controlled setting, with the eventual goal of scaling the approach to continuous-control racing environments like Assetto Corsa, and ultimately to real-world autonomous platforms like the PAVE go-kart.

5. Why CartPole? A Minimalist Racing Proxy

Although high-speed autonomous racing environments like Assetto Corsa offer realism, they are computationally expensive and hard to prototype in quickly. For this reason, we turn to the well-known `CartPole-v1` environment as a lightweight, interpretable domain to test our safety-aware RL pipeline.

CartPole can be interpreted as a simplified racing problem: the pole angle corresponds to dynamic stability, while the cart's horizontal position mirrors track adherence. A failure (i.e., pole falls or cart exceeds track limits) terminates the episode—analogous to going off-track or spinning out in a racing scenario.

This minimalist environment enables us to test whether safety constraints can meaningfully guide learning, even in discrete-action, low-dimensional settings. Crucially, it allows rapid iteration while preserving key characteristics of the racing domain: continuous control requirements, high sensitivity to unsafe behavior, and a narrow performance margin between success and failure.

6. Implementation Details

We begin with a standard Soft Actor-Critic (SAC) implementation and augment it with a Least-Restrictive Safety Filter (LRSF), inspired by Hamilton–Jacobi (HJ) reachability theory. This filter acts as a runtime shield, rejecting unsafe actions proposed by the policy network based on the learned safety Q-function.

6.1. Safety Q-Function

The safety Q-function is trained using Bellman-style backups, but instead of predicting cumulative reward, it estimates the minimum margin to safety over time. Actions that maintain $Q(x,u) > 0$ are considered safe; others are rejected and replaced by a fallback policy.

6.2. Least-Restrictive Safety Filtering

In our CartPole implementation, we adopt a **least-restrictive safety filter** (LRSF) to enforce safety constraints during both training and evaluation. This design follows the reachability-based framework outlined by Borquez et al. [1], where the nominal policy is minimally modified only when the system approaches the boundary of the safe set.

Formally, our learned safety critic $Q^{\text{safe}}(s,a)$ estimates the discounted minimum safety margin for taking action a at state s. We define an action as *safe* if $Q^{\text{safe}}(s,a) > 0$ and *unsafe* otherwise.

The least-restrictive filtering logic is:

$$a_{\text{exec}} = \begin{cases} a_{\text{nominal}}, & \text{if } Q^{\text{safe}}(s, a_{\text{nominal}}) > 0, \\ \arg\max_{a' \in \mathbb{A}} Q^{\text{safe}}(s,a'), & \text{otherwise,} \end{cases}$$

where $a_{\text{nominal}} \sim \pi(\cdot|s)$ is the action sampled from the stochastic SAC policy.

Thus, if the nominal SAC policy proposes an action that maintains positive safety margin, it is executed directly. Otherwise, the filter intervenes and selects the fallback action that maximizes Q^{safe}, ensuring the agent remains within the safe set.

Connection to Reachability Theory. In Hamilton–Jacobi (HJ) reachability analysis, the backward reachable tube (BRT) characterizes states from which failure is inevitable unless a corrective control input is applied. Following Borquez et al., the least-restrictive filter uses the nominal policy freely when safely inside the BRT complement, and only intervenes when necessary near the safety boundary.

Our LRSF mirrors this philosophy:

- If $Q^{\text{safe}}(s,a) > 0$, no intervention is needed and the nominal action is executed.
- If $Q^{\text{safe}}(s,a) \leq 0$, the fallback action maximizing the predicted safety margin is selected instead.

This architecture allows the agent to prioritize task performance while guaranteeing constraint satisfaction with minimal intervention. Over time, as learning progresses, the agent is expected to naturally propose increasingly safe actions, reducing the reliance on fallback corrections.

6.3. Training Setup

The safety margin function $g(x)$ is defined to enforce two constraints simultaneously (pulled from CartPole termination condition documentation):

- The cart position x must remain within ± 2.4 units.
- The pole angle θ must remain within $\pm 12^\circ$ (approximately ± 0.209 radians).

At each state, the instantaneous safety margin is computed as:

$$g(x) = \min\left(2.4 - |x|, \frac{\text{deg2rad}(12) - |\theta|}{5}\right),$$

where x and θ are extracted from the environment's four-dimensional state vector.

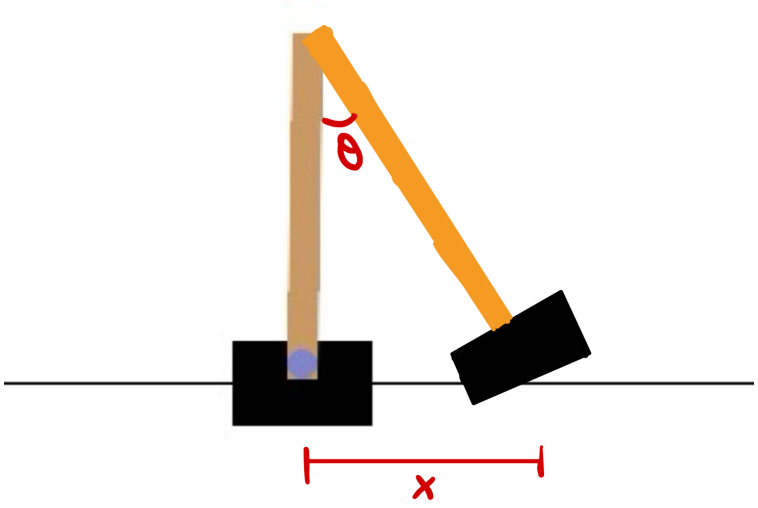

Figure 2: CartPole set-up

This formulation ensures that safety violations can arise either from excessive cart displacement or from pole angle deviation, whichever constraint is more restrictive at a given moment. The division by 5 on the pole angle term acts as a scaling factor to tighten the angular constraint relative to the cart constraint, reflecting the system's increased sensitivity to pole instability.

During both training and evaluation, the Least-Restrictive Safety Filter (LRSF) monitors actions proposed by the SAC policy:

- If the action maintains $Q^{\text{safe}}(s,a) > 0$, it is executed.
- Otherwise, the action is overridden by the safest available action maximizing Q^{safe}.

We train the SAC agent for 1000 episodes, logging both the return trajectory and the frequency of safety interventions. Through this setup, the agent gradually learns to propose actions that autonomously respect the safety constraints, reducing reliance on the safety filter over time.

7. Results and Analysis

7.1. Performance Comparison

As shown in Figure 3, the safety-aware agent achieves consistently higher average return than the baseline SAC agent. While the baseline occasionally spikes to high rewards, it frequently suffers

from early termination due to constraint violations. The safety-aware agent avoids these pitfalls by terminating only when $Q(x,a) \leq 0$ and learning to stay within safe bounds.

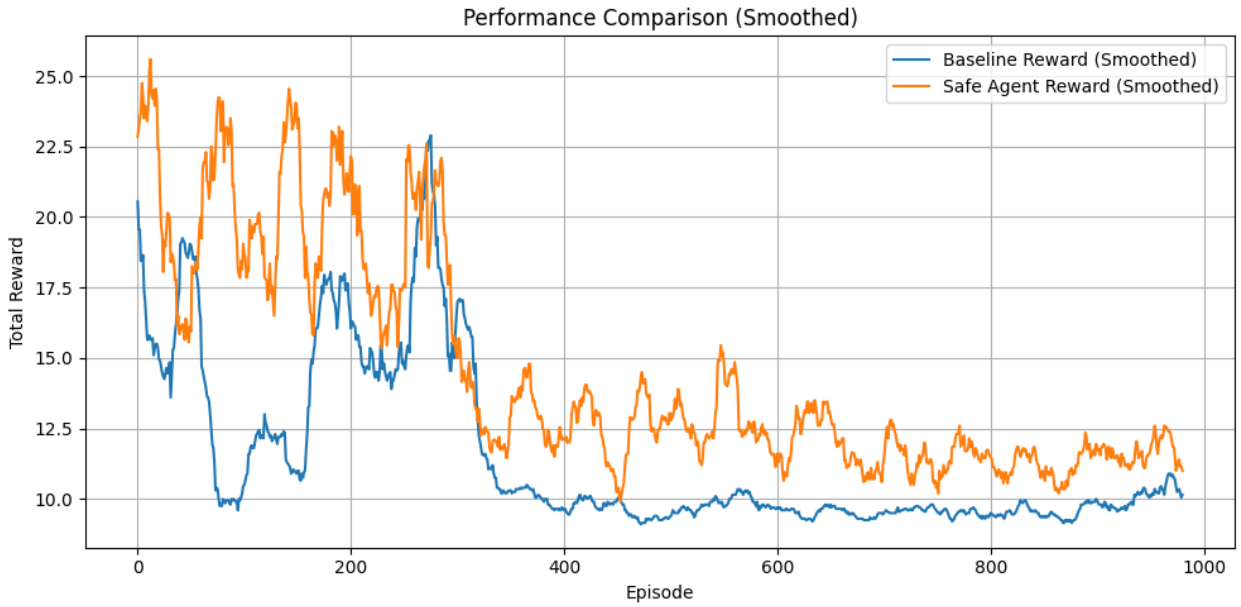

Figure 3: Performance Comparison for Baseline Agent and Safe Agent

7.2. Intervention Breakdown

Figure 4 shows the frequency of safety filter activations across training. The fallback policy dominates early training, but the trusted actions slowly increase over time. This indicates that the agent is learning to select safe actions directly, reducing reliance on the filter.

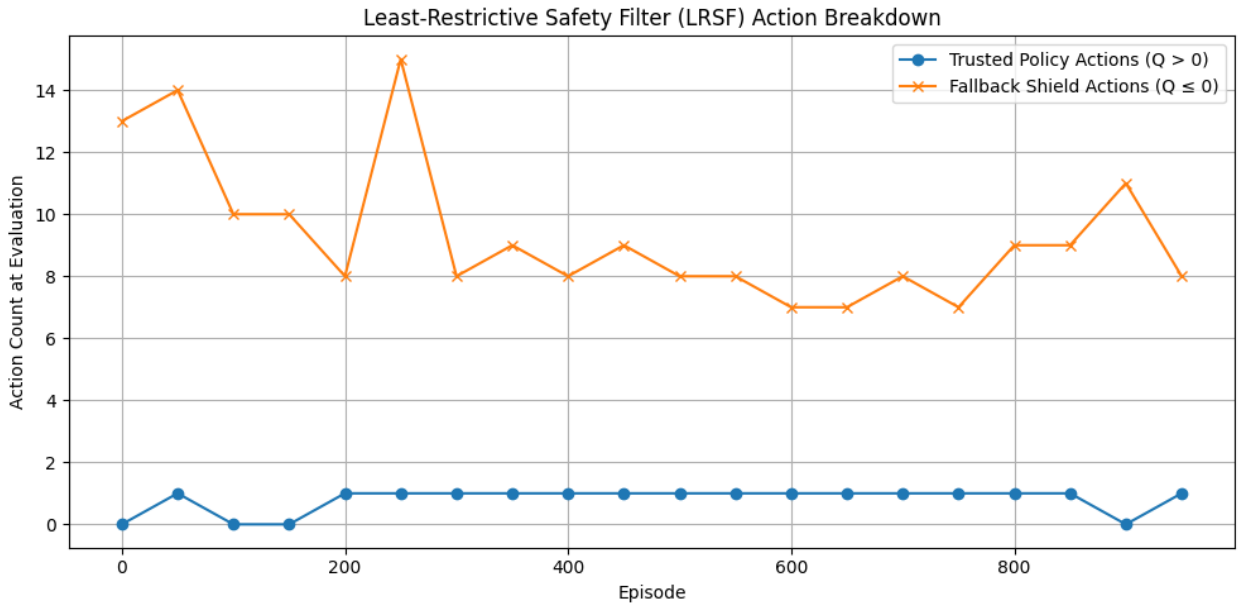

Figure 4: Frequency of Action Intervention by LRSF

7.3. Takeaways

The results of this experiment highlight several key insights into the role of safety-aware learning. First, regarding the **reward versus safety tradeoff**, we observe that incorporating a safety filter does not merely preserve task performance, but actually improves it. By preemptively avoiding catastrophic failures that would terminate episodes early, the safety-aware agent is able to accumulate higher average rewards compared to the unconstrained baseline. This finding challenges the conventional belief that safety enforcement necessarily limits agent performance, instead showing that proper safety integration can enhance stability and long-term returns. Further testing would be required to confidently prove that these safe agents truly perform better (using different random seeds, replay buffer sizes, etc.), but this is a promising starting point.

Second, we find that the **Least-Restrictive Safety Filter (LRSF)** provides substantial utility even when invoked frequently during early stages of training. By shielding the agent from unsafe decisions, the filter regularizes exploration, forcing the policy to learn within safer subregions of the state-action space. This effect leads to faster and more stable convergence by preventing the

destabilizing influence of early unsafe behavior—a critical property for training reinforcement learning agents in safety-critical settings. In

Finally, these results provide initial **proof of scalability** for the learned safety filter architecture. Although the CartPole domain is relatively low-dimensional, the structure of the safety critic and the runtime action filtering mechanism are easily extensible to more complex continuous-control environments. The initial success of this simple implementation suggests that least-restrictive learned safety filters could serve as lightweight yet effective stabilizers and constraint enforcers across a range of high-risk reinforcement learning tasks, including autonomous driving, racing, and robotics.

8. Future Work: From Sim to Track

The long-term goal is to translate this architecture to the domain it was originally motivated by: high-speed autonomous racing. As such, the aim is to:

- **Assetto Corsa Integration**: Test the same LRSF pipeline in the Assetto Corsa simulator using a continuous control SAC agent (e.g., Remonda et al. [9]).
- **Sim2Real Transfer**: Deploy trained safety-aware policies on the Princeton Autonomous Vehicle Engineering (PAVE) go-kart.
- **Human-Centered Filtering**: Incorporate ideas from Human-Centered Safety Filters (HCSF) to preserve driver agency while enforcing safety constraints.
- **Learning from Experts**: Leverage expert demonstration data from racing games or real-world F1 drivers to inform both the task policy and the safety Q-function. Fortunately, comprehensive telemetry data (including speeds, throttle, gear, etc.) from previous F1 races is publicly available [2] (as seen in Figure 5).

This roadmap advances toward real-world autonomous systems that are not only high-performing but inherently safe—without requiring exact models of their dynamics. All of these next steps will be explored in the coming academic year through continued work with the Safe Robotics Lab.

Figure 5: Sample telemetry F1 data

9. Conclusion

This work presents a proof-of-concept implementation of a least-restrictive safety filter (LRSF) for reinforcement learning, applied in the `CartPole-v1` environment. Building on recent advances in safe RL and Hamilton–Jacobi reachability theory, this work demonstrates how safety value functions can be integrated into standard policy optimization pipelines, enabling runtime constraint satisfaction with minimal interference to task performance.

Through the adoption of a discounted safety Bellman formulation, a safety critic is constructed that enables filtering unsafe actions while preserving SAC's stochastic exploration benefits. The results indicate that safety-aware agents can not only maintain constraints reliably but also achieve improved long-term returns by avoiding catastrophic failures that terminate episodes prematurely. This challenges the notion that safety necessarily trades off against performance, and provides evidence that carefully designed safety architectures can in fact accelerate learning in high-risk environments.

Moreover, the modular design of the safety filter—separating task reward maximization from

safety constraint enforcement—demonstrates scalability to more complex, continuous-control domains. Although the experiments here are conducted in a low-dimensional environment, the underlying principles generalize naturally to larger systems, such as autonomous racing simulators and real-world vehicles.

This project serves as an initial stepping stone toward the broader goal of developing safe, high-performance reinforcement learning agents for autonomous racing. Future work will build on these insights to tackle challenges in higher-dimensional settings, integrate expert demonstrations, and transfer learned safety policies onto physical platforms through the PAVE go-kart initiative. By embedding safety into the learning process itself, we move closer to realizing real-world autonomous systems that are not only effective but verifiably safe.

Acknowledgments

I would like to sincerely thank Professor Jaime Fernández Fisac for his guidance, feedback, and mentorship throughout the course of this project. I am also grateful to Donggeon David Oh for his technical advice, encouragement, and support, particularly in developing the safety filter implementation. This project would not have been possible without the research environment provided by the Safe Robotics Laboratory at Princeton University. Finally, I would like to acknowledge Princeton Autonomous Vehicle Engineering (PAVE) for inspiring this work through their Ivy: Racer initiative, which continues to push the boundaries of autonomous racing research at the undergraduate level.

References

[1] J. Borquez, K. Chakraborty, H. Wang, and S. Bansal, "On safety and liveness filtering using hamilton-jacobi reachability analysis," *arXiv preprint arXiv:2312.15347*, 2024.

[2] F1 Tempo, "F1 tempo: F1 telemetry data and analytics," https://www.f1-tempo.com, accessed: 2024-04-26.

[3] J. F. Fisac, N. F. Lugovoy, V. Rubiés-Royo, S. Ghosh, and C. J. Tomlin, "Bridging hamilton-jacobi safety analysis and reinforcement learning," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2019, pp. 8550–8556.

[4] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 2018, pp. 1861–1870.

[5] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[6] D. D. Oh, J. Lidard, H. Hu, H. Sinhmar, E. Lazarski, D. Gopinath, E. S. Sumner, J. A. DeCastro, G. Rosman, N. E. Leonard, and J. Fisac, "Safety with agency: Human-centered safety filter with application to ai-assisted motorsports," in *Robotics: Science and Systems (RSS)*, 2025, to appear.

[7] OpenAI, "Openai gym: Cartpole-v1 environment," https://www.gymlibrary.dev/environments/classic_control/cart_pole/, accessed: 2024-05-01.

[8] Princeton Autonomous Vehicle Engineering, "Princeton autonomous vehicle engineering (pave)," https://blogs.princeton.edu/pave/, accessed: 2024-04-26.

[9] A. Remonda, N. Hansen, A. Raji, N. Musiu, M. Bertogna, E. E. Veas, and X. Wang, "A simulation benchmark for autonomous racing with large-scale human data," in *Proceedings of the Thirty-Eighth Conference on Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2024.

[10] C. J. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3-4, pp. 279–292, 1992.