Autonomous Racecar Navigation with Reinforcement Learning

Mohammad Jaminur Islam
CS258-RL Final Project

1 Introduction

Reinforcement learning (RL) has emerged as a powerful paradigm for training agents to make sequential decisions in complex environments. This project focuses on applying RL techniques to train an autonomous racecar agent to navigate a racetrack. The environment, derived from the `gymnasium` library, offers both continuous action spaces and image-based observations, making it a challenging and realistic benchmark for testing the efficacy of RL algorithms.

The agent must learn to complete laps efficiently, staying on the track while minimizing deviation. Two algorithms, Proximal Policy Optimization (PPO) and Twin Delayed Deep Deterministic Policy Gradient (TD3), were implemented to evaluate their respective strengths and weaknesses in this domain. PPO, being an on-policy algorithm, was expected to achieve stable performance through iterative updates to the policy network. TD3, an off-policy algorithm, was tested for its potential to leverage replay buffers for greater sample efficiency.

This report delves into the design choices, challenges, and experimental results, presenting a comparative evaluation of these algorithms. Section 2 describes the environment and its modifications, Section 3 outlines the model architecture and training configurations, Section 4 discusses the results, and Section 5 reflects on the findings, limitations, and future directions.

2 Environment Setup

The project uses the `CarRacing-v3` environment, which is specifically designed for continuous control tasks. The environment requires the agent to process high-dimensional image inputs and learn to map these inputs to appropriate steering, acceleration, and braking actions. However, achieving this in practice requires modifications to facilitate the agent's learning process:

- **Reduced Lap Completion:** To simplify the task during early stages, the lap completion requirement was reduced to 25%. This gradual increase in task complexity, akin to curriculum learning, helps the agent avoid being overwhelmed by high-dimensional state and action spaces.
- **Observation Aggregation:** A custom environment wrapper was implemented to stack 4 consecutive frames using a `deque`. This provided the agent with temporal

context, allowing it to learn velocity and direction changes, which are critical for decision-making in dynamic environments.

- **Reward Shaping:** A combination of positive rewards for lap completion and penalties for off-track behavior was used. This reward structure serves as a guiding signal, encouraging the agent to follow the track while discouraging erratic exploration.
- **Early Termination:** Episodes were terminated early if the agent's cumulative reward became negative for 100 consecutive steps. This mechanism prevented the agent from learning poor policies and reduced computational overhead by discarding unproductive episodes.
- **Observation Preprocessing:** To streamline the input processing, RGB frames were converted to grayscale, resized to 96×96, and normalized. These steps significantly reduced computational complexity while preserving essential spatial information.

3 Architecture and Parameters

The underlying architecture was designed to balance computational efficiency and representational capacity. Both PPO and TD3 shared a convolutional neural network (CNN) backbone, which extracted features from the preprocessed image inputs.

3.1 Common Network Architecture

The CNN backbone consisted of six convolutional layers with kernel sizes progressively tuned to capture both local and global features. The convolutional layers were followed by fully connected layers that produced a flattened feature vector of size 256. This feature vector was then mapped to actions through additional layers in the policy network (for PPO) or the actor network (for TD3). Continuous actions, including steering and throttle, were scaled between $[-1, 1]$ to match the environment's requirements.

3.2 Proximal Policy Optimization (PPO)

PPO, an on-policy algorithm, updates the policy network by directly interacting with the environment. Its hyperparameters were carefully tuned to achieve stable and efficient learning:

- **Advantage Estimation:** The advantage $A(t)$ was computed as the difference between the returns-to-go $G(t)$ and the value function $V(t)$. The returns-to-go $G(t)$ were calculated recursively as:

$$G(t) = r(t) + \gamma \cdot G(t+1),$$

  where $G(t+1) = 0$ for terminal states. This formulation balances immediate and long-term rewards, enabling the agent to optimize its trajectory over time.
- **Entropy Regularization:** An initial entropy coefficient of 0.02 encouraged exploration by penalizing deterministic policies. This coefficient decayed to 0.001, allowing the agent to converge to stable policies as training progressed.

- **Optimization Setup:** Training involved 3 policy epochs and 5 value epochs per iteration, with a batch size of 256 and a learning rate of 5×10^{-4} for the policy network.

3.3 Twin Delayed Deep Deterministic Policy Gradient (TD3)

TD3, an off-policy algorithm, leverages replay buffers to sample past experiences for training. Key design choices included:

- **Policy Noise:** Gaussian noise with a standard deviation of 0.1 was added to policy outputs during training. Noise clipping ensured stability by bounding the noise within a fixed range.
- **Replay Buffer:** A buffer storing up to 10,000 experiences facilitated efficient sampling and reduced overfitting.
- **Training Frequency:** The policy network was updated every two steps, while the value network was updated at every iteration. This delayed policy update stabilized training.
- **Optimization Setup:** Learning rates were set to 7×10^{-4} for the actor network and 1×10^{-4} for the critic network.

4 Results

Experiments revealed distinct learning characteristics for PPO and TD3. PPO demonstrated faster and more consistent improvement in rewards, achieving a peak score of 597. Early instability, caused by insufficient state representations, was resolved by aggregating frames and shaping rewards. The curriculum-based approach also contributed to this success. In contrast, TD3 struggled to achieve similar performance, indicating its sensitivity to hyperparameter tuning and its reliance on high-quality samples from the replay buffer. Despite extensive experimentation, TD3's rewards remained inconsistent, highlighting the challenges of off-policy learning in high-dimensional environments.

5 Discussion and Future Directions

Proximal Policy Optimization (PPO): PPO's success underscores the importance of stable and well-tuned policy updates. The inclusion of entropy regularization enabled the agent to explore diverse strategies, avoiding premature convergence to suboptimal policies. The results highlight PPO's ability to adapt to complex control tasks, especially when paired with curriculum learning and temporal aggregation techniques.

Twin Delayed Deep Deterministic Policy Gradient (TD3): TD3's inconsistent performance reflects the inherent challenges of off-policy training, particularly in environments with high-dimensional observations. The reliance on a replay buffer introduces additional complexities, such as the risk of stale samples and overestimation bias. Future work could explore integrating memory-based architectures, such as recurrent neural networks (RNNs) or Transformers, to address these issues.

Future Work:

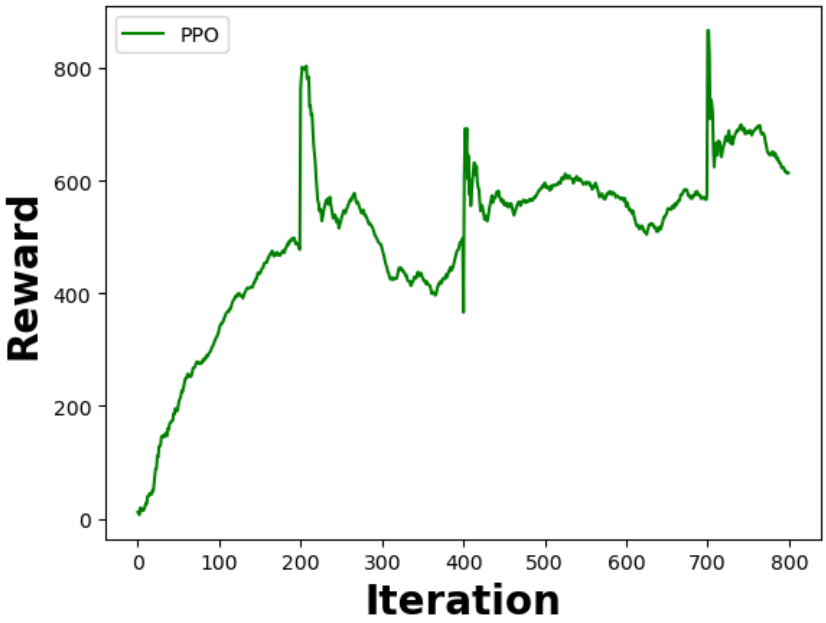

Figure 1: Rewards over iteration for PPO

- **Improved Architectures:** Incorporate sequential models to enhance temporal reasoning and improve decision-making capabilities.
- **Advanced Curriculum Learning:** Gradually increase lap completion requirements to enable agents to handle full laps over extended training periods.
- **Scalable Training:** Deploy training on GPU clusters to accelerate convergence and enable testing of larger architectures.

6 Conclusion

This project highlights the potential of reinforcement learning algorithms for autonomous navigation tasks. PPO proved effective in navigating the racetrack, leveraging its robustness and adaptability to achieve high rewards. While TD3 faced challenges, its potential for future improvements remains promising. Continued research in this domain could pave the way for more efficient and capable RL-based navigation systems.