

STAT 4830

Reinforcement Learning Racing Agent

Second Weekly Report: Training Iterations and Model Refinements

Team: Matthew Lobo, Yuv Malik, Pablo Echevarria Cuesta

Project Overview

We are training an autonomous racing agent using reinforcement learning. The agent learns to drive entirely from visual input—a 96×96 RGB image representing its view of the track—and outputs continuous control signals for steering, acceleration, and braking.

The environment (*multi_car_racing*) provides realistic physics simulation and a reward signal based on track progress. Our goal is to optimize the agent's policy to complete laps quickly while avoiding crashes and staying on the track.

PPO

ALGORITHM

1M

TIMESTEPS

96×96

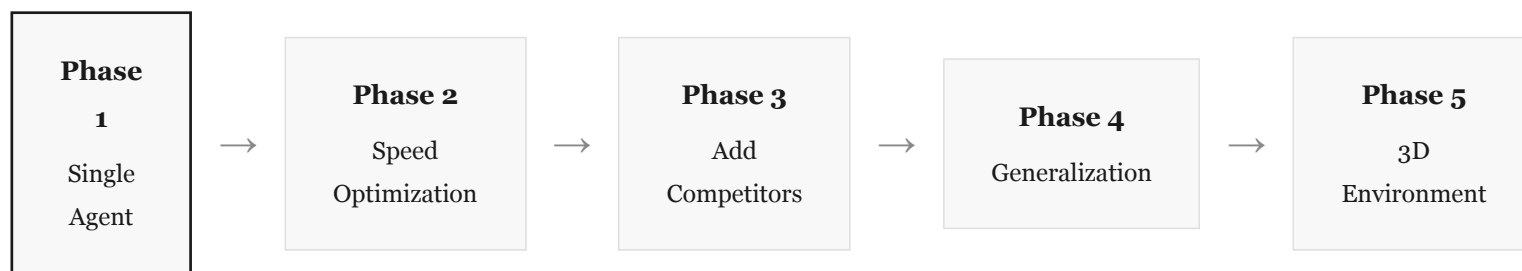
RGB INPUT

$[-1, 1]^3$

ACTION SPACE

Project Roadmap

This project develops in stages. We started with a single agent learning to drive on one track. Our goal is to eventually build a general-purpose racing AI that can compete against opponents on any track—and ultimately operate in 3D environments.



CURRENT FOCUS

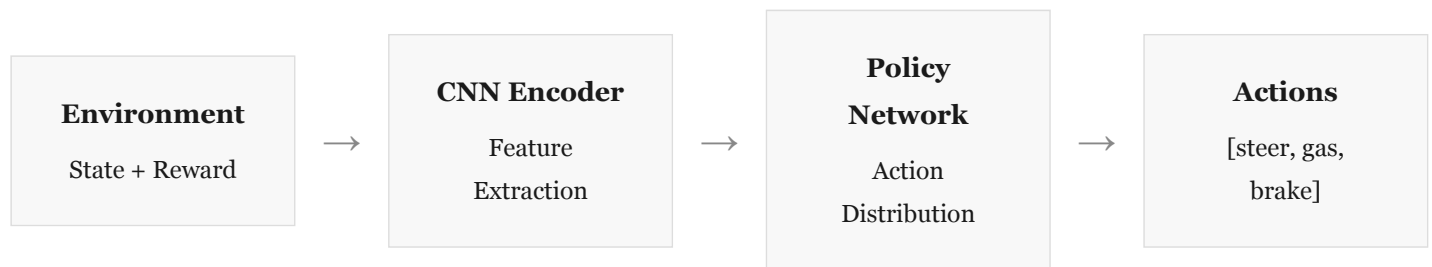
We are in **Phase 2**. The agent can complete laps, but drives too slowly. This report covers our reward shaping experiments and the challenges of balancing speed with safety.

END GOAL

A racing agent that generalizes across tracks, competes against multiple opponents, and adapts to 3D environments with realistic physics—trained entirely through reinforcement learning.

Training Architecture

We use Proximal Policy Optimization (PPO), a policy gradient method that updates the policy while constraining how much it can change per iteration. This provides stable learning for continuous control tasks. The policy network uses convolutional layers to extract spatial features from the image input.



Batch size: 1024 **Rollout:** 1024 steps **Learning rate:** 3×10^{-4} **Discount (γ):** 0.999

Parallel envs: 8

Baseline Training Results

Initial training over 500K timesteps with default hyperparameters showed that the agent could learn basic driving behavior. Mean episode reward improved from -61 to 282+, demonstrating that the policy was converging toward useful navigation behavior.

LEARNED BEHAVIORS

- Maintains forward motion along track
- Follows track boundaries
- Completes full laps consistently
- Episodes reach max length (1000 steps)

IDENTIFIED PROBLEMS

- Excessive speed leading to corner crashes
- No braking behavior before turns
- Off-track driving to exploit shortcuts
- Speed prioritized over control

Issue 1: Speed Overweighting

The agent learned to maximize speed at all costs because the base reward function implicitly favors velocity. Track progress is measured by tiles visited per step—faster movement means more tiles, which means higher cumulative reward.

$$R_{\text{base}} = (\text{tiles visited}) \times (1000 / \text{track length}) - 0.1 \times (\text{time steps})$$

The time penalty of -0.1 per step is too small to counterbalance the reward from aggressive driving. The agent discovers that high-risk, high-speed strategies yield better expected returns than cautious driving—even accounting for occasional crashes.

Issue 2: Sharp Turn Handling

The agent had no mechanism for anticipating curves. It would enter sharp turns at full speed and either crash or lose control. To address this, we implemented curvature-based reward shaping that encourages braking when approaching high-curvature track segments.

```
# Detect sharp turns by measuring track curvature ahead curvature =  
measure_track_curvature(lookahead=6) is_sharp_turn = (curvature > 0.35) # radians #  
Reward braking only when needed (approaching turn while fast) if is_sharp_turn and speed  
> 5.0: brake_reward = min(brake_action × 0.4, 0.5) # capped per step
```

This conditional reward avoids penalizing speed globally. The agent only receives the braking bonus when it's both approaching a sharp turn and moving fast enough that braking is appropriate.

Issue 3: Off-Track Exploitation

The agent discovered it could cut corners by driving on grass, bypassing difficult track sections entirely. Without a strong disincentive, off-track driving became a viable exploitation strategy.

SOLUTION: TERMINAL OFF-TRACK PENALTY

We made off-track events terminate the episode immediately with a large negative reward. This creates a clear, unambiguous learning signal that leaving the track is catastrophic—not just suboptimal.

```
if car_on_grass: reward += -1.0 # per-step penalty while off-track reward +=  
-100.0 # terminal penalty done = True # episode ends immediately
```


Reward Function Design

Our modified reward function combines the environment's base reward with additional shaping terms. The total reward at each step is:

$$R_{\text{total}} = R_{\text{progress}} - R_{\text{time}} + R_{\text{brake}} - R_{\text{steer}} - R_{\text{off-track}}$$

POSITIVE SIGNALS

- **Progress:** +1000/L per new tile visited
- **Braking:** +0.4 × brake action (on sharp turns)
- **Centerline bonus:** +0.1 to +0.5 for optimal position

NEGATIVE SIGNALS

- **Time:** -0.1 per step
- **Steering jitter:** -0.05 × |Δsteer|
- **Off-track:** -1.0/step, -100 terminal

Current Challenge

Our reward shaping successfully reduced crashes and off-track events, but introduced an unintended side effect: the agent now drives slowly everywhere. Instead of learning context-dependent speed control, it learned that slow driving is universally safe.

OBSERVED BEHAVIOR

- Consistently low velocity on all track sections (straights and curves)
- No speed differentiation based on track geometry
- Braking reward may be encouraging unnecessary caution
- Poor lap times despite high completion rate

This illustrates a fundamental challenge in reward shaping: local incentives produce global behavioral effects. The braking reward, intended only for corners, shifted the entire policy toward conservative driving.

Lessons Learned

- **Reward functions define learned behavior:** The agent optimizes exactly what we reward. Implicit biases in the reward (like tile-based progress) lead to unintended strategies (like speed maximization).
- **Local shaping has global effects:** Adding a braking reward for sharp turns made the agent cautious everywhere, not just on corners. Reward terms interact in non-obvious ways.
- **Terminal penalties provide strong gradients:** Making off-track events end the episode gave the clearest learning signal. The agent quickly learned to avoid the track boundary.
- **Balance is difficult:** Over-penalizing risky behavior produces overly conservative policies. Under-penalizing allows exploitation. The optimal reward function is narrow and problem-specific.

Next Steps: Velocity-Based Rewards

To fix the slow-car problem, we will redesign the reward to explicitly encourage efficient forward movement rather than just track progress. The key insight is to reward velocity in the track direction, not tiles visited.

```
# Replace tile-based progress with directional velocity reward R_velocity =  
dot(car_velocity, track_direction) # Increase time penalty to create urgency  
time_penalty = -0.5 # was -0.1 # Remove explicit brake reward (prevents point farming) #  
Instead: Speed-to-Curvature penalty safe_speed = f(curvature) # lower for sharp turns if  
current_speed > safe_speed: penalty = -(current_speed - safe_speed) × 0.3 # Scale off-  
track penalty by speed off_track_penalty = -(1.0 + speed × 0.5)
```

Additional Strategies

CURRICULUM LEARNING

Begin training on tracks with gentle curves, then progressively introduce sharper turns. This allows the agent to master speed control incrementally rather than facing the full complexity from the start.

CONTEXT-AWARE SPEED TARGETS

Define optimal speed profiles for each track segment based on curvature. Reward the agent for matching target speeds: fast on straights, slow on corners—encoding the behavior we want explicitly.

VELOCITY DECOMPOSITION

Separately reward forward velocity (along track) and penalize lateral velocity (sliding). This encourages controlled cornering while maintaining speed through optimal racing lines.

MULTI-OBJECTIVE OPTIMIZATION

Treat speed and safety as separate objectives. Use Pareto optimization techniques to explore the trade-off frontier rather than collapsing everything into a single scalar reward.

Project Roadmap

Beyond solving the speed optimization problem, we have a broader vision for this project. Our current agent drives alone on a single track—but real racing involves competitors, varied environments, and eventually, more complex 3D dynamics.

PHASE 1: ADD COMPETITORS

Introduce other cars on the track. The agent must learn not just to drive fast, but to navigate around opponents, avoid collisions, and find opportunities to overtake. This transforms the problem from single-agent to multi-agent RL.

PHASE 2: GENERALIZATION

Train a policy that works on any track layout with any number of competitors. This requires the agent to learn generalizable driving skills rather than memorizing a single environment—a significant step toward robust racing AI.

LONG-TERM: 3D RACING ENVIRONMENT

Eventually, we want to move to a 3D racing simulator. This introduces new challenges that our current 2D setup doesn't capture:

- **Elevation changes:** Hills, slopes, and banking affect vehicle dynamics and visibility
- **3D observation space:** Larger visual input, more complex feature extraction
- **Realistic physics:** Weight transfer, tire grip, aerodynamics become factors
- **Computational cost:** Higher-dimensional state space requires more training resources

Summary

This week we focused on reward shaping to address behavioral issues in our racing agent. We successfully eliminated crashes and off-track exploitation through curvature-based braking rewards and terminal penalties. However, these modifications introduced a new problem: overly conservative driving—which motivates our next phase of work.

ACCOMPLISHED

- Stable lap completion without crashes
- Consistent track boundary adherence
- Appropriate braking on sharp turns
- Clear understanding of reward dynamics

WHAT'S NEXT

- Velocity-based reward redesign
- Introduce competing agents
- Generalize across tracks and opponents
- Explore 3D environment transition

Goal: A general-purpose racing agent that competes, adapts, and wins.