

# Bonus Project – CarRacing-v3

## 1. Introduction & Problem Overview

This bonus project was designed to extend our reinforcement learning experiments to a different environment and control setting.

We chose the CarRacing-v3 environment from Gymnasium, a well-known benchmark where an autonomous vehicle must learn to drive on randomly generated tracks using only pixel observations as input.

The main challenge lies in learning continuous control, managing steering, throttle, and braking simultaneously from high-dimensional visual data without any prior knowledge of physics or driving rules.

The agent is rewarded for moving forward and penalized for going off-track, forcing it to balance speed and stability

To address this, we implemented and trained a Proximal Policy Optimization (PPO) agent capable of directly mapping sequences of frames to continuous control signals.

The objective was to achieve a stable driving policy that could consistently complete laps with high efficiency.

Beyond the technical purpose, this project also included a fun experimental component with the creation of a “Human vs AI” race mode, where a player could directly compete against the trained agent on the same track, turning research into an interactive experience.

## 2. Environment & Reward Function

Link: [https://gymnasium.farama.org/environments/box2d/car\\_racing/](https://gymnasium.farama.org/environments/box2d/car_racing/)

The CarRacing-v3 environment from Gymnasium provides a top-down 2D view of a car driving on procedurally generated racing tracks.

At each timestep, the agent receives an RGB image of size 96×96 pixels, representing its current field of view.

The goal is to complete as much of the track as possible while minimizing time off the road.

### Action Space (Continuous):

The control is continuous, defined by a 3-dimensional action vector:

- Steering  $\in [-1, 1]$ : controls left and right turns.
- Gas  $\in [0, 1]$ : accelerates the car forward.
- Brake  $\in [0, 1]$ : slows the car down.

This continuous formulation allows smoother and more realistic driving behaviors compared to discrete control, avoiding abrupt directional changes.

### Observation Space:

The observation consists of pixel-based visual input, which is then:

- Converted to grayscale to reduce dimensionality.
- Stacked over 4 consecutive frames to capture short-term temporal dynamics (velocity, direction, etc.).

This gives the agent a richer representation of motion without requiring explicit state information.

### Reward Function:

The agent gains a positive reward for every new section of track visited and loses points when it slows down or leaves the road.

Episodes end early when the car stops making progress, encouraging steady forward motion and penalizing instability.

## **3. Algorithm & Training Setup**

We used Proximal Policy Optimization (PPO) implemented in PyTorch.

After testing a discrete-action PPO version, which reached high scores but suffered from large variance (up to -120 after +900 runs), we switched to continuous control, which offered smoother steering and better stability.

### Model Architecture:

- Convolutional neural network (CNN) encoder for visual input.
- Two heads: a Gaussian policy (actor) and a scalar value function (critic).
- Optimization with Adam, using GAE-lambda for stable advantage estimation.

### Training Strategy:

The model was trained for 6 million steps, divided into four distinct phases governed by adaptive hyperparameter scheduling:

| Phase | Steps  | Learning Rate | Entropy Coef | Rollout Horizon |
|-------|--------|---------------|--------------|-----------------|
| 1     | 0-1.5M | 5e-5          | 1e-3         | 4096            |
| 2     | 1.5-3M | 3e-5          | 7e-4         | 3072            |
| 3     | 3-4.5M | 1.5e-5        | 4e-4         | 2048            |
| 4     | 4.5-6M | 5e-6          | 1e-4         | 1024            |

This gradual annealing strategy helped balance exploration and exploitation, maintaining high entropy in early stages for broader search, then reducing it to stabilize policy behavior. The learning rate decay ensured smoother convergence and reduced oscillations in late-stage optimization.

Training was logged in real-time using a live visualization tool (live\_plot.py), showing reward progression, moving averages, and learning rate evolution. Models were automatically checkpointed whenever the average reward reached a new high.

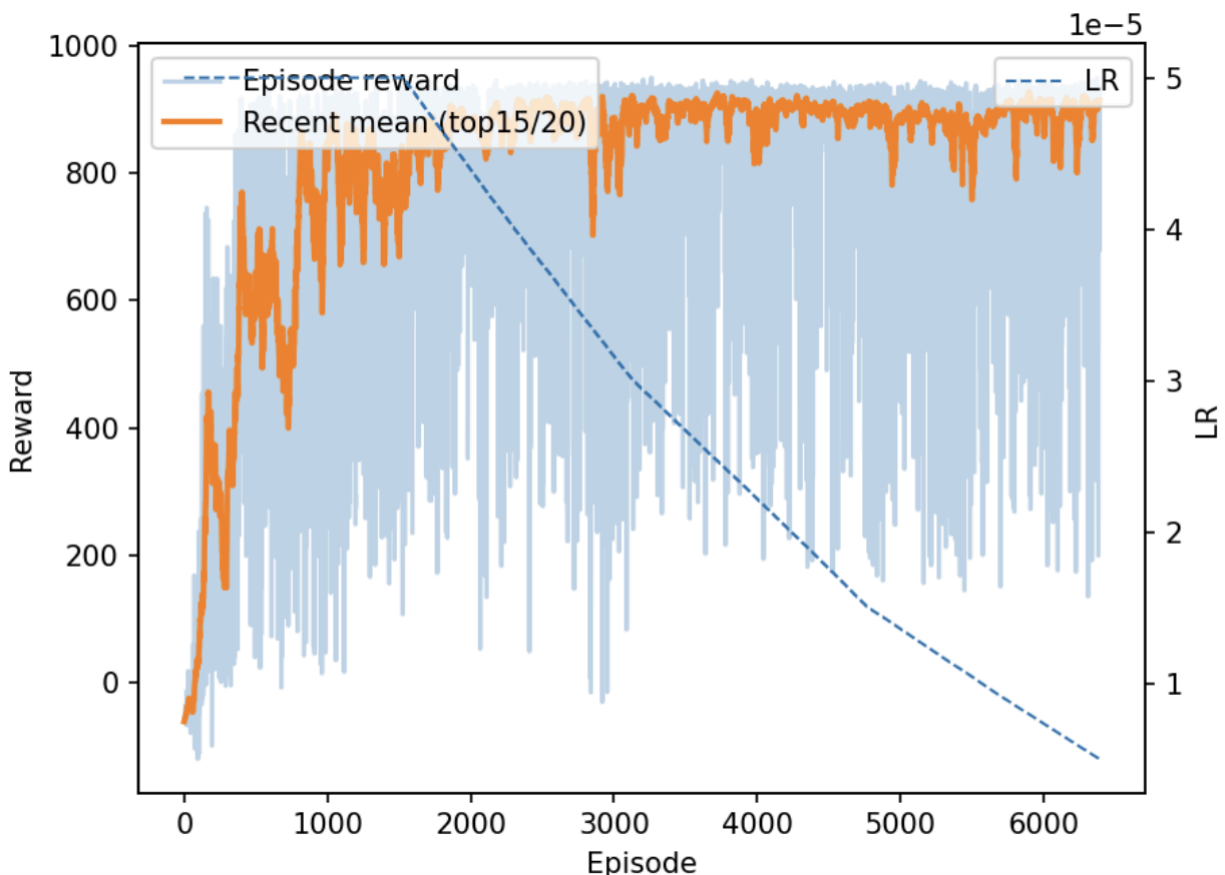
## 4. Experiments & Results

After six million training steps, the continuous PPO agent achieved average rewards around 900, with peaks near 950.

Performance became more stable over time as entropy decreased, reflecting the agent's shift from exploration to mastery.

The reward curve showed a steady improvement, with the moving average staying above 900 during late training.

Failures, typically when the car spun off-road, became rare after phase 3.



During evaluation, the agent displayed fluid and human-like control, maintaining lane position, braking smoothly before turns, and accelerating out of corners. It was able to finish laps reliably on most tracks, even with random generation.

Typical late-stage log:

```
[final4 ep 6378] rew=884.08 recent_mean=910.12 steps=5990074 lr=5.1e-06 ent=0.0001 horz=1024 phase=3
[final4 ep 6379] rew=867.85 recent_mean=905.21 steps=5991074 lr=5.1e-06 ent=0.0001 horz=1024 phase=3
[final4 ep 6380] rew=900.90 recent_mean=906.83 steps=5992065 lr=5.1e-06 ent=0.0001 horz=1024 phase=3
[final4 ep 6381] rew=200.00 recent_mean=906.83 steps=5993065 lr=5.1e-06 ent=0.0001 horz=1024 phase=3
[final4 ep 6382] rew=877.85 recent_mean=903.87 steps=5994065 lr=5e-06 ent=0.0001 horz=1024 phase=3
[final4 ep 6383] rew=934.70 recent_mean=908.59 steps=5994718 lr=5e-06 ent=0.0001 horz=1024 phase=3
[final4 ep 6384] rew=890.85 recent_mean=910.12 steps=5995718 lr=5e-06 ent=0.0001 horz=1024 phase=3
[final4 ep 6385] rew=928.00 recent_mean=913.47 steps=5996438 lr=5e-06 ent=0.0001 horz=1024 phase=3
[final4 ep 6386] rew=919.40 recent_mean=913.21 steps=5997244 lr=5e-06 ent=0.0001 horz=1024 phase=3
[final4 ep 6387] rew=677.78 recent_mean=911.45 steps=5998244 lr=5e-06 ent=0.0001 horz=1024 phase=3
[final4 ep 6388] rew=919.50 recent_mean=913.83 steps=5999049 lr=5e-06 ent=0.0001 horz=1024 phase=3
❖ FINAL PHASE CONTINUOUS TRAIN DONE
```

## 5. Analysis of Performance, Observations, and Conclusion

The final PPO model achieved stable and generalizable driving behavior:

- **Consistency:** Regular scores above 900 demonstrated strong policy stability.
- **Realism:** Continuous actions led to natural driving behavior with fine control.
- **Generalization:** The policy adapted well to unseen tracks, not just memorized layouts.
- **Limitation:** Recovery from off-road situations remains weak due to PPO's lack of memory.

The 4-phase scheduler proved critical: it kept early exploration broad and gradually locked in optimal behavior, resulting in smoother driving and faster convergence.

The Human vs AI mode added a creative dimension. A split-screen race where the trained agent competes directly with a player.

The AI generally outperformed casual players, showcasing learned precision and control while keeping the game engaging and competitive.

In conclusion, this bonus experiment confirmed the robustness of PPO for continuous, vision-based tasks.


Despite being an add-on to our main Pong work, it evolved into a complete and enjoyable demonstration of how reinforcement learning can produce a capable, lifelike driver — able to race, adapt, and challenge humans 🏁.

## 6. Team & Roles

This CarRacing project started as a side experiment while our main focus was on the Pong environment.

Originally meant as a backup, it became a successful and entertaining bonus demonstration.

- **Arnaud Fernandes** – Reinforcement Learning Lead (Car Racing)



Focused on the CarRacing PPO model: implemented the continuous-action agent, tuned hyperparameters, managed training schedules, and created the “Human vs AI” mode.

- **Yacine Lassouani** – Pong Model & Experimentation Support

Led the main Pong training, assisted in debugging and optimizing the CarRacing agent, and helped align both projects under a consistent PPO framework.

- **Valentin Gorse** – Design, Documentation & Presentation

Contributed ideas for the Human vs AI setup and handled report structuring and presentation design, ensuring clarity and visual consistency.

Together, we built two reinforcement learning agents. One mastering Pong, the other mastering CarRacing, combining research, coding, and creativity in a fun and applied way.