Autonomus Robot for the Detection of Landmines in a Simulated Environment

Jhojan Stiven Aragon Ramirez
Dept. of Computer Engineering
Universidad Distrital Francisco José de Caldas
Email: jhsaragonr@udistrital.edu.co

Kevin Emmanuel Tovar Lizarazo
Fundamentos de ciencias y sistemas
Universidad Distrital Francisco José de Caldas
Email: ketovarl@udistrital.edu.co

I. ABSTRACT

II. INTRODUCTION

Autonomous control of vehicles has become a benchmark problem in reinforcement learning, but reproducing real-world driving in the laboratory often requires complex simulators or physical testbeds. In this project, we simplify the task by focusing on the Gymnasium CarRacing-v0 environment [1], where a planar racing car must follow a procedurally generated track using only on-board sensors. The initial problem we address is thus configuring a virtual racing car to drive through a circuit—rather than dealing with real vehicles or full 3D simulators—so that we can concentrate on core challenges of perception, action selection, and reward design in a lightweight Box2D physics setting [1]. By adopting CarRacing-v0, which already handles chassis joints, tire friction, and steering dynamics internally, we avoid reinventing vehicle dynamics and can direct our efforts toward improving the learning algorithm itself.

In particular, several recent works focus on the Gymnasium CarRacing-v0 environment, providing step-by-step DQN implementations and baseline performance metrics that serve as a foundation for further improvements [2]. By analyzing these implementations—including the detailed project report by Perod et al. [2]—we identified opportunities to enhance observation processing and reward shaping.

In this work, we present a DQN-based agent for the CarRacing environment in the Gymnasium library [1]. This environment uses the Box2D physics engine to solve the planar rigid–body equations

$$m\dot{v} = \sum F, \quad I\dot{\omega} = \sum \tau,$$

and handles chassis joints, tire friction, and steering internally. By building on this existing model, we avoid implementing vehicle dynamics from scratch and focus instead on the learning algorithm.

To improve the agent's perception, we augment the standard RGB input with a simulated 14-ray LiDAR sensor mounted at the front of the vehicle. The LiDAR readings provide a vector of distances to nearby obstacles, giving the agent direct information about track geometry before it appears in the camera view. Inspired by prior work on obstacle detection and autonomous driving [**?**], our agent processes both image frames and LiDAR data through parallel neural network branches that merge before the Q-value output layer.

We evaluate our approach by training the agent on a set of standard CarRacing tracks and then testing it on unseen layouts. Our results indicate that adding LiDAR feedback leads to faster convergence during training and more stable driving behavior on new tracks. This suggests that combining visual and range-based observations can help DQN agents generalize better in driving tasks.

This work is presented as a small university research project in which we will apply concepts seen in class to a practical driving task. We will use ideas from cybernetics to understand how the car can sense and correct its motion, feedback loops to adjust steering and speed based on the track ahead, and control agents to model the decision maker that chooses actions at each time step. Our main contribution is to build a DQN agent that combines image input and a simulated 14-ray LiDAR sensor, showing that this multimodal approach helps the car learn faster and drive more smoothly on new tracks. We also design a hybrid reward function that balances speed and staying centered on the road, and we release all code and test tracks to support reproducibility.

The rest of the document is organized as follows. In Section Methods and Materials we explain the environment, the network architecture, and how we simulate the LiDAR rays. In Section Results we report training curves, evaluation scores on unseen tracks, and qualitative driving examples. Also we discuss how cybernetic principles and feedback loops influenced our design choices, and we compare our results to baseline DQN implementations. Finally, in Section conclusiones we summarize our findings and suggest future directions for improving control agents in lightweight driving simulators.

III. METHODS AND MATERIALS

We began by drawing a component diagram to show how each part of our system fits together (see Fig. 1). This diagram clarifies the relationships between our sensor inputs, the learning agent, and the robot's control outputs before describing the technical details.

Our setup uses the CarRacing-v0 environment from Gymnasium, which provides a two-dimensional, top-down view of a procedurally generated racetrack. At each time step, the agent

Funded by *Your Funding Source* under grant No. 12345.

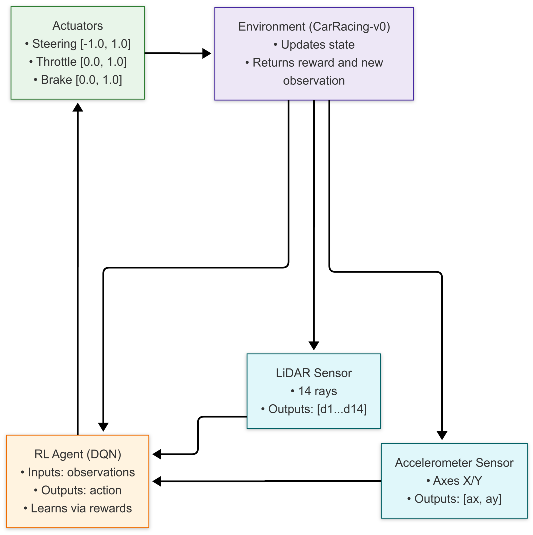

Fig. 1. Component diagram of the system.

receives a single 96×96 RGB image showing the car at the center of the track. In addition to this visual input, Gymnasium exposes four ABS sensor readings (one for each wheel), true speed, steering position, and gyroscope measurements. We combine these data into a single state representation that the agent observes.

In addition to the visual and sensor data provided by the CarRacing-v0 environment, the initial problem formulation also considered the use of LiDAR sensors. We aim to explore the integration of LiDAR, potentially by modifying the existing environment or interfacing with a simulated LiDAR. This additional sensory input can be seamlessly incorporated into the state representation fed to our DQN agent, further enriching its perception of the surroundings.

To navigate and avoid obstacles, we implemented a Deep Q-Network (DQN) agent. The DQN uses a convolutional neural network to approximate the action-value function over continuous control actions—steering, gas, and brake. During training, the agent's experiences (state, action, reward, next state) are stored in a replay buffer. We periodically sample mini-batches from this buffer to update the main network, while a separate target network—updated every few episodes—helps stabilize learning.

The reward function, provided by the Gymnasium CarRacing-v0 environment [1], is designed to encourage both speed and track coverage. At each frame, the agent receives a -0.1 penalty, which pushes it to finish the track quickly. Each newly visited track tile yields a bonus of $1000/N$ points, where N is the total number of tiles. For example, covering all tiles in 732 frames yields:

$$1000 - 0.1 \times 732 \approx 926.8 \text{ points.}$$

Crashing off-track for too long or failing to visit new tiles ends the episode.

Figure 2 illustrates the overall DQN workflow. In our implementation, the input image and sensor readings pass through convolutional and fully connected layers to produce Q-values for each action. The action with the highest Q-value is selected at each step. Over thousands of episodes, the agent learns to balance acceleration, braking, and steering to maximize cumulative reward while avoiding track boundaries.

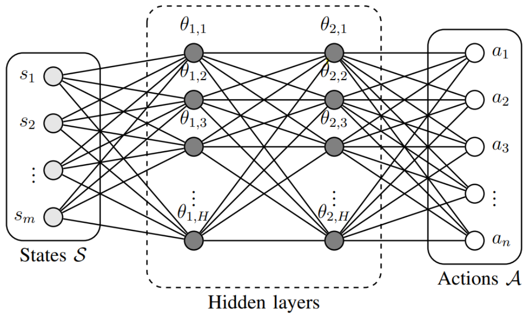

Fig. 2. DQN algorithm diagram [3].

By combining visual information with sensor data and training with the DQN algorithm in the 2-D CarRacing-v0 environment, our system learns to navigate complex tracks efficiently and safely. This approach lays the foundation for future extensions, such as adding more sophisticated perception modules or testing on physical robots.

IV. Results

V. Conclusions

Acknowledgment

VI. bibliography

References

[1] F. Foundation, "Gymnasium carracing environment," 2023. [Online]. Available: https://gymnasium.farama.org/environments/box2d/car_racing/

[2] jperod, "Si final project: Ai self-driving race car using deep reinforcement learning," GitHub repository, 2019, accessed: 15 May 2025. [Online]. Available: https://github.com/jperod/AI-self-driving-race-car-Deep-Reinforcement-Learning/blob/master/SI_Final_Project.pdf

[3] F. Mismar, J. Choi, and B. Evans, "A framework for automated cellular network tuning with reinforcement learning," 08 2018.