{EPITECH}

ROBOCAR

RACING SIMULATOR

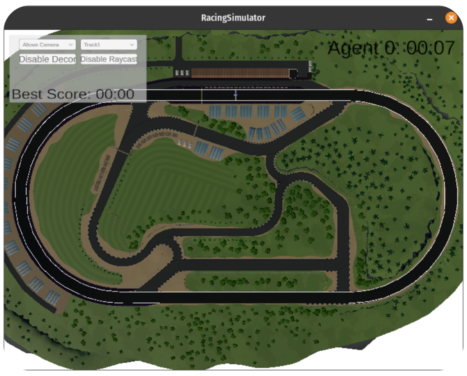
ROBOCAR

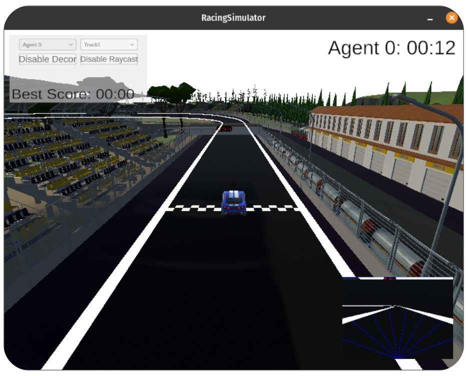
Language: Python

What is Racing Simulator?

The objective of this project is to initiate the **Robocar** project by focusing on the artificial intelligence (AI) aspect.
You will have access to a simulation similar to the one you will work with during the **Robocar** project.
Using this simulation, your goal is to train your AI to navigate the track as quickly and efficiently as possible.

✓ A slow car that stays within the track boundaries will be considered better than a fast car that crosses the lines.





Project Structure

This project is divided into four main components:

- ✓ A **client**
- ✓ An **input management system**
- ✓ A **data collector**
- ✓ An **artificial intelligence**

How to Use the Simulator

Click here to download the simulation

Simply run the binary (**RacingSimulator**).

You will find a menu in the top-left corner of the window, where you can change the view or the track.
There are two buttons that allow you to activate/deactivate the decor or the raycast display to improve the performance of the simulation.

Once set up, you just need to connect your client.

Client

To connect your program to the simulation, you will need to use the mlagent library.
Create a UnityEnvironment, which will open the simulation and connect to port 5004.

To create agents, you need to pass additional parameters. Those should be: –config-path
The JSON file should look like this:

```
{
    "agents": [
        {
            "fov": 180,
            "nbRay": 10
        },
        {
            "fov": 48,
            "nbRay": 36
        }
    ]
}
```

Fov should be between 1 and 180, while number of ray should be between 1 and 50.

Using this configuration, two agents will be connected and controllable through mlagent.

Input Management System

The input management system allows you to control the car.
You need to capture keyboard inputs and send them to the simulation via the server to maneuver the car.

✓ Your input management must be precise since you will use it to navigate the tracks and collect high-quality data.
✓ You can try to use a controller if you can as controlling the car using the keyboard can be challenging.

Data Collector

To train your supervised AI model, you will need data to feed it.
For this, you must create a data collector that records data from **your** driving trajectories.

✓ Ensure your data is clean, usable, and collected in sufficient quantity.
✓ Conduct exploratory data analysis (EDA) to validate data quality.

Artificial Intelligence

✓ Do **not** create a reinforcement learning model!

Start by developing a supervised learning model that learns from the data collected by the data collector.

Important Considerations

✓ Use evaluation metrics, watching the car drive is not enough to determine if the model is good or bad.
✓ The dataset is a really important part, it needs to be big, clean and diverse.
✓ Remember having a small model with a big dataset is way better than a big model with a small dataset.

Bonus Features

If everything works as expected, consider implementing the following improvements, it will help you for the following steps of the project:

- ✓ Model quantization for hardware difference with the jetson nano
- ✓ Advanced trajectory computation
- ✓ Multi-agent learning
- ✓ Reinforcement learning (**consult the project supervisors first**)

v 1.0.0

{EPITECH}