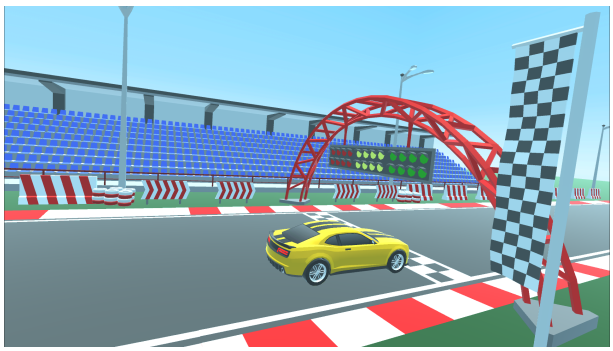

User Manual

Contents

1 Overview

This application enables the training of neuroevolution Agents through a car racing game-like scenario and graphical user interface. Alongside this, an implementation of reinforcement learning is also available to train Agents, however, this must be done outside of the application and requires additional set-up.

If you simply wish to explore Neuroevolution, please follow the Basic Installation instructions. If you wish to train both Neuroevolution and Reinforcement Learning Agents, please follow the Advanced Installation instructions.

Following installation and verifying you can run the application, you can start training your Agents. For instructions on how to train using neuroevolution, see the Training a Neuroevolution Agent instructions in section 3. For instructions on how to train using reinforcement learning, see the Training a Reinforcement Learning Agent instructions in section 5.

2 Basic Installation

This will guide you through the steps to simply download and run the application. This will enable you to train Neuroevolution Agents. If you wish to train Reinforcement Learning Agents or open the project in the Unity Editor, please follow the Advanced Installation instructions in Section 4.

- Note: The application has only been built for Windows and only tested on Windows 11

1. Download the "ApplicationBuild" directory from the repository
2. Locate the Project.exe file and run it
3. The application will now load and you can create and train Neuroevolution Agents

3 Training a Neuroevolution Agent

Training an Agent using Neuroevolution is simple. The following steps will guide you through the process of creating, training, saving and loading your first neuroevolution Agent.

3.1 Creating a new Agent

- Launch the application. You will be met with the home screen, seen in Figure 3.1. Here you have the option to create a new Agent, load a previous Agent or exit the application. To create a new Agent click the **New Agent** button.
- You will be met with the New Agent screen. Here you can access all the parameters used to create your neuroevolution agent. Figure 3.2 shows the New Agent screen with key elements highlighted.
  - In **green** is the current parameter menu you are looking at. Initially, you are looking at the Map and Method parameters. Each of the parameter menus and what the parameters do is listed under **Neuroevolution Parameters** below. You can change this menu by clicking the various buttons highlighted in pink on the left-hand side.
  - If you wish to return to the home screen, click the **Return** button highlighted in **white**.
  - To change the name of your Agent, you can edit the text highlighted in **blue**.
  - Once you have set all the parameters you wish, you can click the **Start** button highlighted in **red**.
- After setting your parameters, Agents name and clicking Start, you will be met with the loading screen, seen in Figure 3.3. Once your model has loaded, you will be prompted to press Enter to begin training. Tap Enter and you will be sent to the training screen and begin training.

Figure 3.1: The home screen

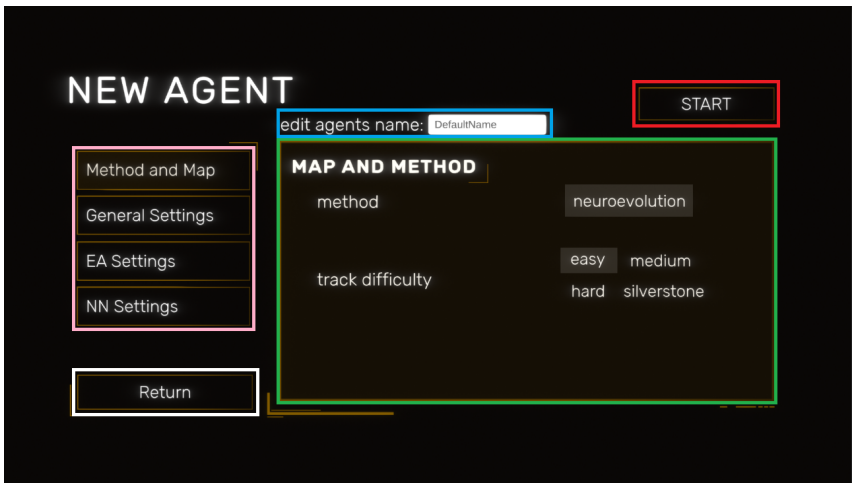

Figure 3.2: The New Agent screen on the Map and Method parameter menu with key elements highlighted

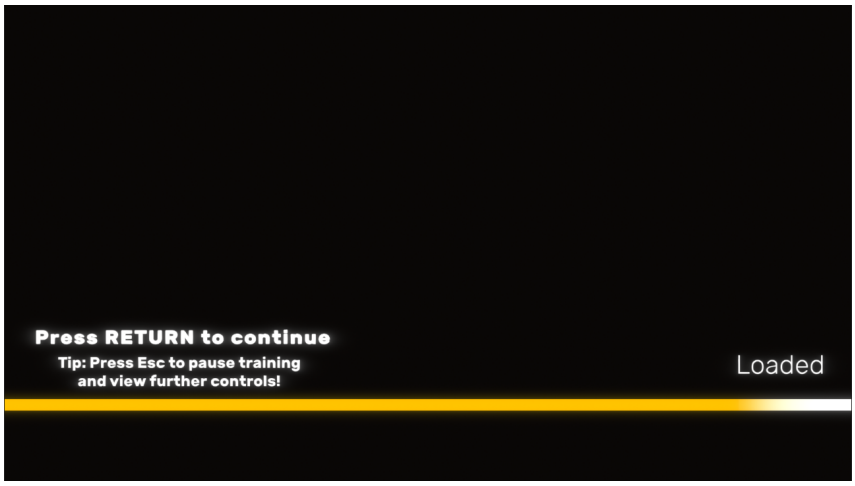

Figure 3.3: The loading screen

3.2 During Training, the Pause Menu and Controls

- Figure 3.4 shows the training screen if the easy map was selected. You can see the racetrack and cars. In the top right is a small overlay displaying the current generation and whether you have random spawn points enabled (more on that later).
- Tap Esc (Escape) to pause training. Here you will be met with the pause menu, seen in Figure 3.5. Here there are buttons to view the parameters you set, return home, quit the application and save your agent alongside a reminder of the controls and their functions.
- During training the following controls can be used:
  - Tap '1' to toggle the neural network graph. This is a visual representation of the best individual's neural network from the previous generation. An example graph can be seen in Figure 3.6.
  - Tap '2' to toggle the fitness graph. This is a graph displaying the best and average fitness of each generation over time. An example graph can be seen in Figure 3.7.
  - Tap '3' to toggle the sensor visualisation. This will visualise the sensors each car uses to tell distance as bright pink lines. An example of this being on can be seen in Figure 3.8.

- Tap '4' to toggle the camera position. By default, the camera is static, looking down on the track. By tapping 4 you can toggle this to follow the best car around the track, or back to static. Figure 3.9 displays what this looks like.
- Tap '5' to toggle random spawn points. When active, each generation of cars will spawn at random points around the track. This can help in generalising the Agent.

- When you are happy with your Agent, or the termination criteria is met, save your model and return to the home screen.

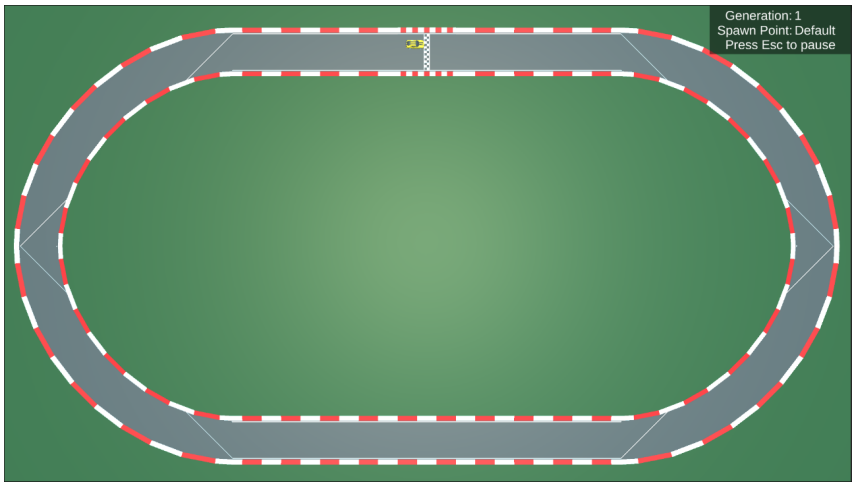

Figure 3.4: The training screen

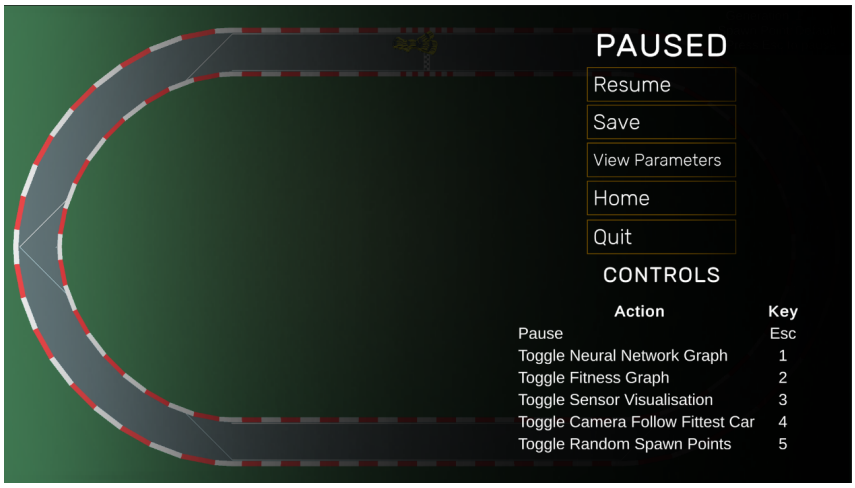

Figure 3.5: The pause menu

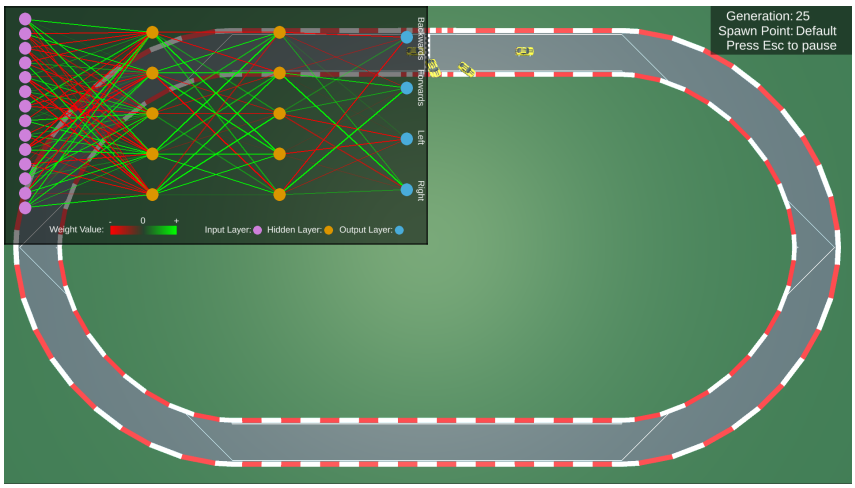

Figure 3.6: An example of the Neural Network Graph

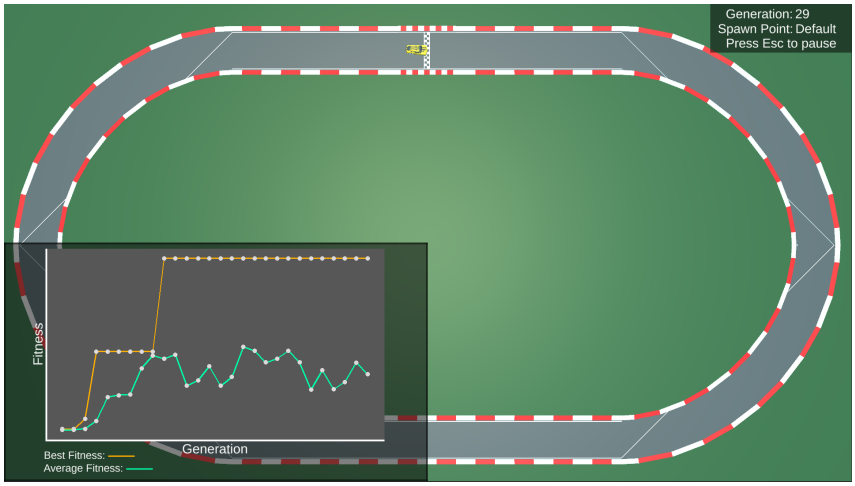

Figure 3.7: An example of the Fitness Graph

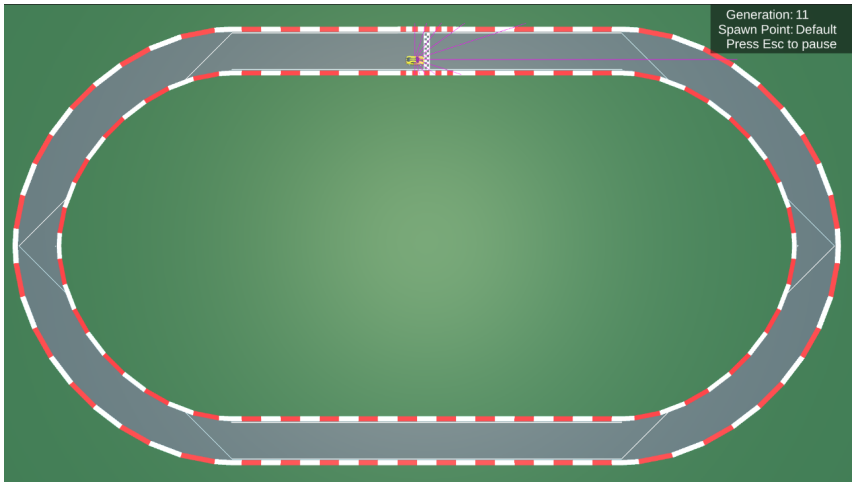

Figure 3.8: An example of having Sensor Visualisation on

Figure 3.9: An example of having the camera follow the fittest car

3.2.1 Quick Controls Reference

Key	Action
Esc (Escape)	Pause
1	Toggle Neural Network Graph
2	Toggle Fitness Graph
3	Toggle Sensor Visualisation
4	Toggle Camera Follow Fittest Car
5	Toggle Random Spawn Points

3.3 Loading a Previously Trained Agent

- On the home screen, click the **Load Agent** button. A menu will appear, as seen in Figure 3.10. Your screen may differ based on the Agents you have trained.
- Every Agent you have saved will appear here. The name you entered when creating the Agent identifies each Agent, alongside a time stamp of when you saved the model. This is in the form of: year-month-day_hour-minute-second.
- Select the map difficulty you would like to continue training on for that Agent. Scroll down the list of saved Agents and find the Agent you want to continue training. Click the **Start** button on that Agent to begin training.

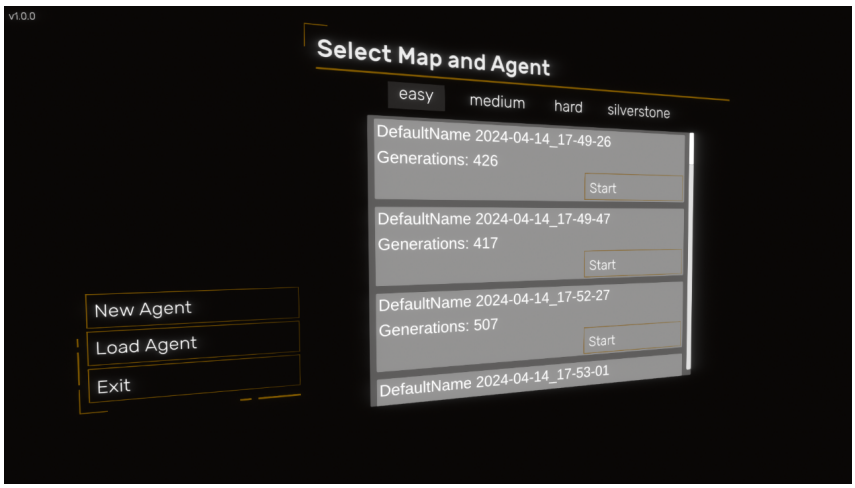

Figure 3.10: The load menu

3.4 Neuroevolution Parameters

- **Map and Method** (Figure 3.2):
  - **method**: There is only one option here, Neuroevolution. If you wish to use reinforcement learning, please refer to the advanced installation instructions (section 4) and the Training a Reinforcement Learning Agent (section 5).
  - **track difficulty**: What track the agent will train on.
- **General Settings** (Figure 3.11):
  - **number of sensors**: The number of distance sensors each car will have and be input into the neural network.
  - **termination condition**: Either Max Time or Max Generation. When this value is reached, training will end.
  - **max time**: The maximum length of time training will continue if Max Time is selected as the termination condition.
  - **max generations**: The maximum number of generations training will continue if Max Generation is selected as the termination condition.
- **EA** (Evolutionary Algorithm) **Settings** (Figure 3.12):
  - **population size**: The number of individuals (cars) in the population.
  - **crossover**: Toggle this On or Off. The type of Crossover. Either Neuron Swap, Weight Swap, Bias Swap or Layer Swap. This refers to how the genetic material of each parent (the neural network of each parent) will be combined to create two new individuals (neural networks).
  - **mutation**: Toggle this On or Off. The type of Mutation. Either Change Value by +/- 0-1, Change Value by 0-200%, Multiply Value by -1 or Completely Replace Value. This refers to how the genetic material of an individual will be mutated. Value refers to the weight or bias of the neural network chosen to be mutated.
  - **parent selection method**: How the parents for crossover will be selected. Either Tournament Selection or Roulette Wheel.
  - **tour size**: Only used if Tournament Selection is chosen as the parent selection method. The number of individuals that will be randomly selected from the population to compete in the tournament.

- **elitism size**: The number of the best individuals in the population to be carried across to the next generation.
- **ssr percent**: Steady State Replacement percent. The percent of the population (after elitism) to fill with offspring.

- **NN** (Neural Network) **Settings** (Figure 3.13):
  - **number of hidden layers**: The number of hidden layers in the neural network. Changing this value will dynamically change the number of layers available to edit.
  - **edit layers**: Each hidden layer of your neural network.
    - **number of nodes**: The number of nodes in this hidden layer.
    - **activation function**: The activation function for each node in this layer. Either ReLU, Tanh, Sigmoid or None (linear).

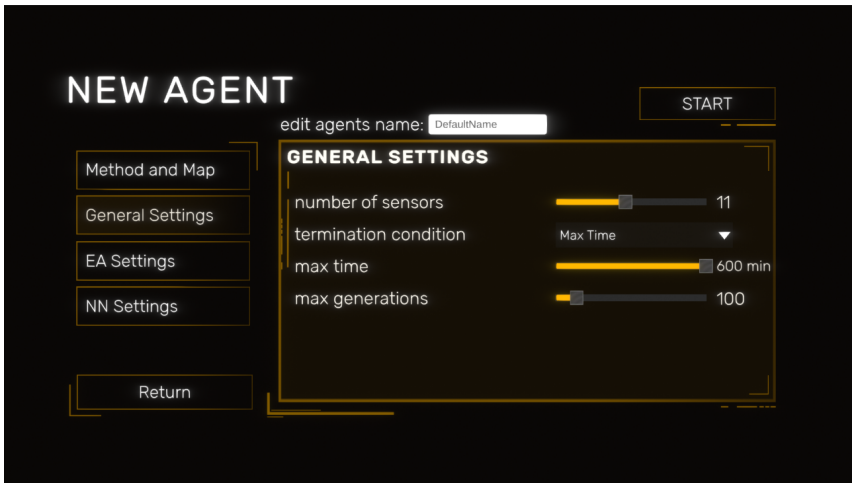

Figure 3.11: The New Agent Screen on the General Settings parameter menu

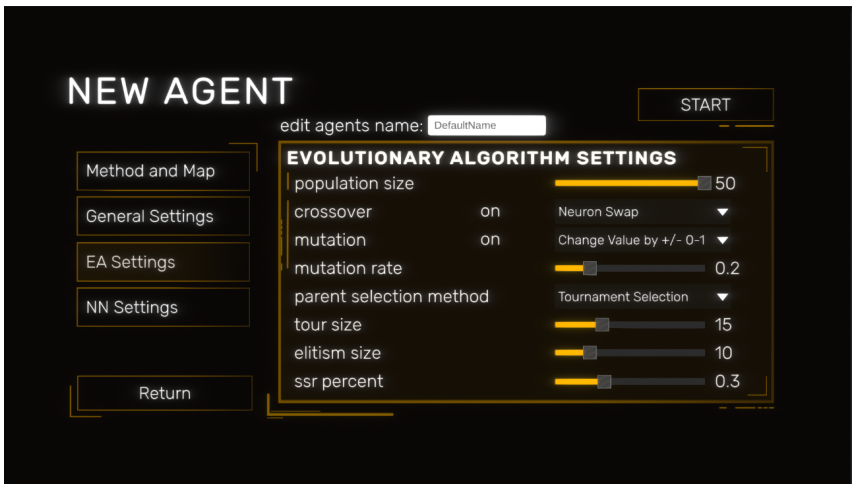

Figure 3.12: The New Agent Screen on the Evolutionary Algorithm Settings parameter menu

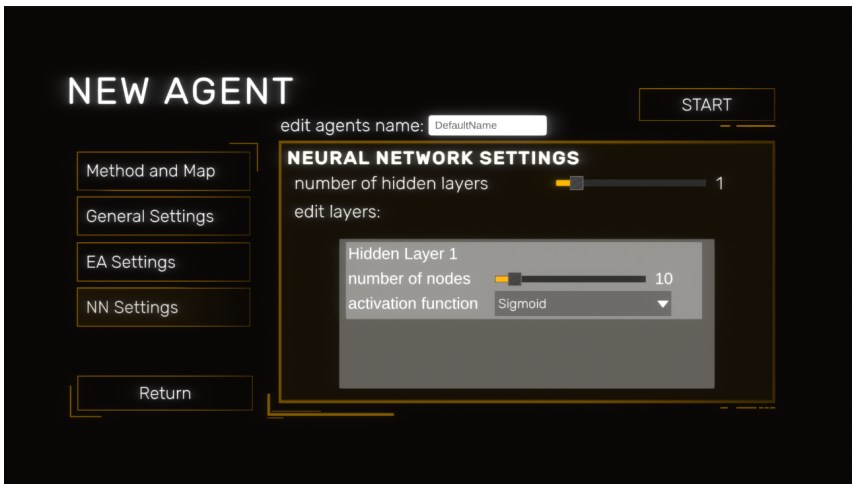

Figure 3.13: The New Agent Screen on the Neural Network Settings parameter menu

4 Advanced Installation

This will guide you through the process of installing if you wish to open the project in the Unity Editor or if you wish to run Reinforcement Learning.

- Note: This has only been tested on Unity Editor version: 2022.3.10f1. Other versions of the editor may cause issues.
- Note: ML-Agents release 20 is the only tested version. Other versions of ML-Agents may not work.

1. Download the repository
2. (Optional, only necessary if you wish to edit the project) Add the project to Unity Hub. Click Add, From Disk, select the "Project" folder
    - An error concerning ML-Agents will appear, click continue
    - An error concerning compilation will appear, click ignore
    - The project should now load into the Unity editor with compilation errors related to mlagents. This is because we have not yet installed ML-Agents
3. - Follow the installation instructions for ML-Agents. Only version 20 has been tested
    - - This will involve installing Python, only Python version 3.9.13 has been tested
4. - Once ML-Agents is installed, in the terminal navigate to the Project directory
5. - Activate your Python environment
6. - The following command will begin training using the pre-built environments:

```
mlagents-learn config/car_config.yml --env=*track*RL --time-scale=15 --width=1280
--height=720 --run-id=MyRunID
```

- Replace '**MyRunID**' with the name of your Agent.
- **width** and **height** specify the size of the environment window. Feel free to change this depending on your screen size or preference.
- **time-scale** specifies the speed at which the Unity environment will run at. A higher time-scale will result in faster training, although it is not advised to go above a value of 20 as Unity's physics engine can become unstable.
- This command assumes you are in the 'Project' directory. If you are not, you may have to edit the path to the config file and the env.
- The config file is located within the Project folder at Project/config/car_config.yml. This file contains the parameters used in training. Feel free to edit the parameters. For further detail on these or other available parameters visit the ML-Agents GitLab or web docs page.

5 Training a Reinforcement Learning Agent

Training a reinforcement learning Agent is slightly different to training a neuroevolution Agent as we use ML-Agents, a machine learning package by Unity, to handle the reinforcement learning algorithm. The following instructions will guide you through the process of training your first Agent with ML-Agents, however for more detail on the config file, command line options or anything else, visit the ML-Agents GitLab page or their web docs:

- Assuming you have followed the Advanced Installation instructions, open a command prompt. Navigate to the 'Project' directory.
- Activate your Python virtual environment.
- The following command will begin training. An environment will open and the reinforcement learning algorithm will start. **Important**: Depending on which track difficulty you would like to train on, replace *track* in the command with either 'Easy', 'Medium', 'Hard' or 'Silverstone'. Also note:
  - Replace '**MyRunID**' with the name of your Agent.
  - **width** and **height** specify the size of the environment window. Feel free to change this depending on your screen size or preference.
  - **time-scale** specifies the speed at which the Unity environment will run at. A higher time-scale will result in faster training, although it is not advised to go above a value of 20 as Unity's physics engine can become unstable.
  - This command assumes you are in the 'Project' directory. If you are not, you may have to edit the path to the config file and the env.
  - The config file is located within the Project folder at Project/config/car_config.yml. This file contains the parameters used in training. Feel free to edit the parameters. For further detail on these or other available parameters visit the ML-Agents GitLab or web docs page.

```
mlagents-learn config/car_config.yml --env=*track*RL --time-scale=15 --width=1280
--height=720 --run-id=MyRunID
```