# INSTRUCTIONS

Project should be opened using *Unity 2019.2.17f1*

## AI Object

A screenshot of a cell phone

Description automatically generated

Acceleration Value  
Exists to display the value the bot is outputting for acceleration.

Turning Value  
Exists to display the turning value the bot is outputting for turning.

Overall Fitness  
Displays the fitness value of the current agent. (Higher is more important)

Average Speed Multiplier  
Tweak this to change the importance of the agent’s average speed in fitness calculation. (Higher is more important)

Sensor Multiplier  
Tweak this to change the importance of the agent staying central on the track. (Higher is more important)

Distance to Target Multiplier  
Tweak this to change the importance of the agent reaching the defined target object. (Higher is more important)

Distance from Start Multiplier  
Tweak this to change the importance of the agent being further from the defined starting location. (Higher is more important)

LAYERS  
Change this number to change the number of hidden layers.

NEURONS  
Change this number to change the number of neurons per hidden layer.

Normalise Sensor Value  
These help the sensors stay between 0 and 1 in value, do not change.

## Neural Network Tweaking Help

The higher the neuron and hidden layer count, the more nuanced the decisions made by the neural network can be.

The lower the neuron and hidden layer count, the faster the AI can be trained and the more likely that random populations of the neural network are to be right. The network can make less nuanced decisions, however.

The fitness values can be tweaked using the multiplier values, allowing for the user to define which elements are more important for the agents to learn.

## GeneticController Object

A screenshot of a cell phone

Description automatically generated

Initial Population  
This changes the number of agents per generation. Anything above 50 is fine but the lower the number the higher the chance of getting an index\_out\_of\_bounds error within the genetic controller.

Mutation Rate  
Tweak this to change the rate of mutation after the crossover/breeding process

Best AI Selection  
Decide how many of the highest performing AI’s get to breed together. This helps the agents improve.

Worst AI Selection  
Define how many of the lowest performing AI’s get to be bred, this helps with the genetic variation of the agents over time.

Number to Crossover  
The number to crossover is how many agents within a generation can breed.

Current Generation  
A display of the current generation

Current Genome  
A display of the current AI Genome being simulated.

Skipping an Agent  
A currently simulating agent can be skipped by pressing the left ctrl key.

## Genetic Algorithm Tweaking Help

Changing the initial population setting to a lower value allows for generational breeding to happen more regularly, but allows for less variation at each generational step. The lower the value, the more likely an out of bounds error, this is a bug which I have yet to figure out.

Change the mutation rate to a higher value, the float is a likelihood of mutation after breeding.

Changing the Crossover Controls section allows you to tune the agents so that they are less likely to become dead set in certain behaviours.

## Making the Simulation Faster

The simulation speed can be increased by changing the time scale in project settings.

*Go to Edit -> Project Settings -> Time -> Time Scale*

Due to each agent being simulated individually, this setting can be cranked up high to offset the fact that only one agent is being simulated at a time.

A lower initial population setting can also allow the agents to learn new behaviour faster. A balance between the complexity of the neural networks and the numbers of agents per generation needs to be found in order for training to happen within a desired timeframe.

## Future Improvements

In order to improve this project, there are some desired features that can be added.

The first feature would be the ability to save an agent as a JSON file for use in storing it for use in other scenarios. The ability to load a neural network would also be desired, with the ability to determine the number of hidden layers and neurons used so that they are not mismatched.

The ability to run multiple agents at a single time would also be desirable, however as the project currently stands the ability to manipulate the speed of the situation allows for any shortcomings of simulating a single agent to be mitigated.

The ability to change the number of sensors that an agent can make use of would help towards creating more efficient agents, as currently the project is hard coded to use only three inputs.

## Link To Video

YouTube:   
<https://www.youtube.com/watch?v=xrgjvl1YkEM>

Google Drive: <https://drive.google.com/file/d/1JtW7yZHTxR5v0C9wKqzSu3l8blooynFI/view?usp=sharing>

## References

This project is built as a modified version of the following tutorial:  
<https://www.youtube.com/watch?v=C6SZUU8XQQ0>

The video submitted alongside this project makes use of a single royalty free sound:  
<https://www.bensound.com/royalty-free-music/track/retro-soul?sort=p.sort&order=DESC>

The project makes use of the Math Net for matrix functions:  
<https://numerics.mathdotnet.com/>