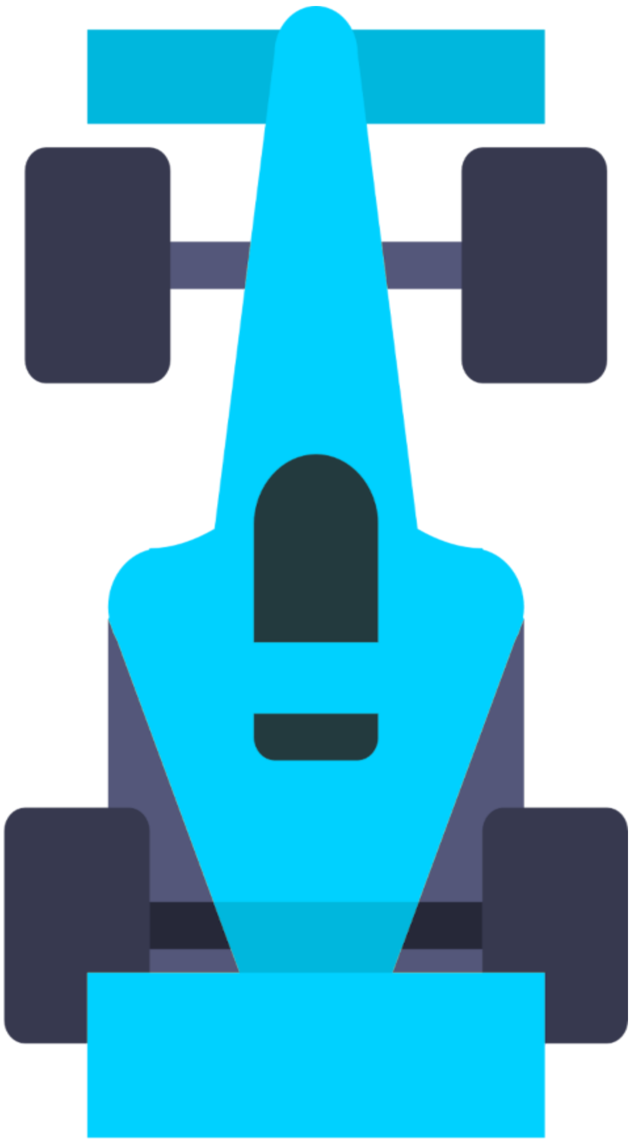
The Effect of Population Size on the Number of Generations it Takes for at Least One Car to Complete a Lap



— Madison Wedding —

Question: How does population size influence the number of generations it takes for at least one car to complete a lap?

Aim: To determine the effect of the starting population size on the number of generations it takes for at least one car to complete a lap.

Hypothesis: If population size affects the number of generations it takes for at least one car to complete a lap, then having a larger population will decrease the number of generations needed, because each generation will have more cars that are mutating and therefore will have more beneficial mutations that will allow them to learn to complete the track faster.

Variables:

Independent variable: Population size
Population size refers to the number of cars which start alive at the beginning of each generation. It can be changed in line 4 of the config.txt file, by simply changing the value of the variable pop_size.

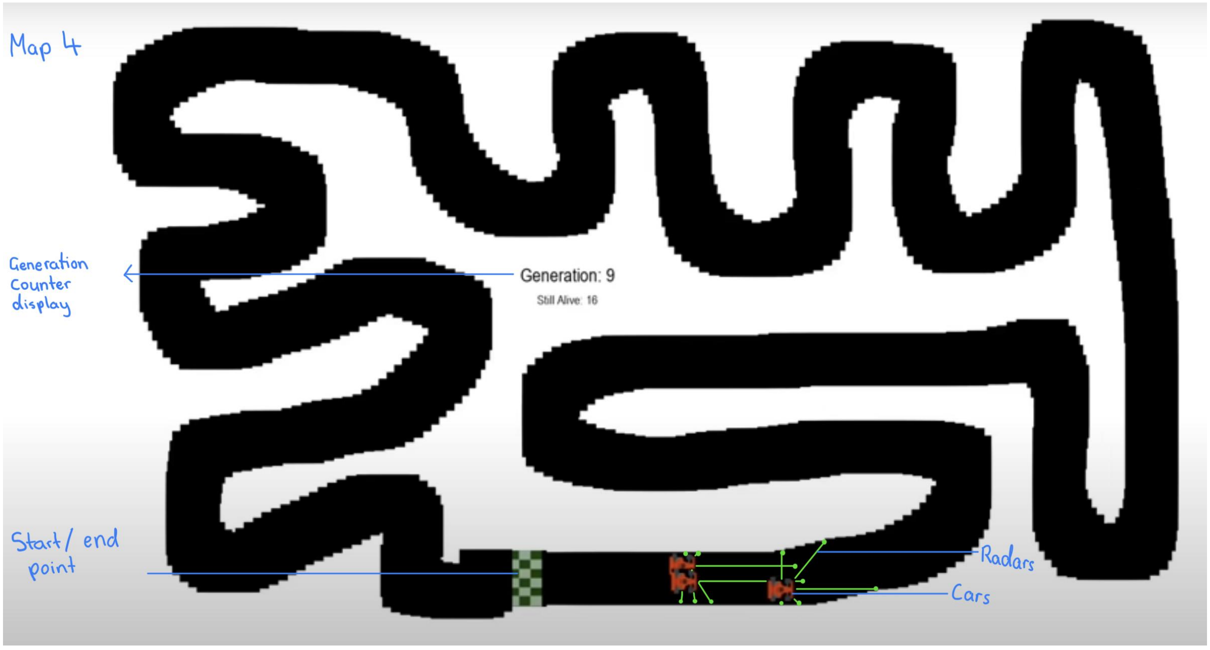
Dependent variable: The number of generations for at least one car to complete a lap
I will be noting how many generations there are until the cars have progressed enough for at least one of them to be able to complete a full lap of the track. It will be measured using the current_generation variable, which counts how many generations of cars there has been at any point throughout the simulation, and by visually seeing when the first lap by a car has been completed (ie. when it returns to its starting position).

Controlled Variables:

Variable	What is it?	How will it be controlled?
Map (Map 4)	The map refers to the path that the cars have to follow. They vary in difficulty, meaning that they have a massive effect on how many generations it will take for the first car to complete a lap. If the map is harder, it will take longer and if the map is easier, it will take less generations.	This variable is quite easy to control, as the map is chosen in line 268 of the code, using the line: `game_map = pygame.image.load("map4.png").convert()` As long as I do not change the map image chosen, this variable will be controlled.

Car Size	The size of the car needs to remain the same throughout all trials, because if the car is in a different ratio to the track, it will have more/ less space within the borders. If it has less space, it will be more challenging for the car to progress because it will always be closer to the edges, increasing the chance of collisions. Likewise, if the car was smaller, it would have more space to move on the track, meaning that it would be harder for it to crash.	To change this variable would again require manual work, so as long as I do not change the code, this will stay the same.
Some Behaviours of the cars	Sometimes, the cars decide to stop moving or go around the track backwards. When it stops moving, the time value that contributes to the overall fitness is very high, increasing the chance of cars in future generations mimicking the strategy. When cars use this strategy, they are not overly likely to complete a lap, since they choose to sit in a corner until they are timed out, which would greatly affect my results. If cars go around the track backwards, they are essentially completing a different track, which may vary the difficulty, which would also invalidate the experiment.	If any of the cars start to display any of these behaviours, I will shut down the simulation and restart it. I will do this for every trial in which one or more cars do so, so that this variable can be controlled.
The rest of the code and configuration in general (newcar.py and config.txt)	The rest of the code must remain the same in the experiment, because it affects the way that the cars learn and move. If some cars are learning through a different algorithm than others, it may be more or less effective, which would affect my results.	This would take purposeful hindrance to deem it uncontrolled, so as long as I do not make any changes to the python file, or the config file (besides changing the population size), this variable will be controlled.
Amount running on my computer	If I had a lot of programs running at once, my computer may be slowed down, which could have an effect on the cars.	To keep the amount running on my computer minimal, I will only run vscode, and the window with the simulation. Everything else, I will close, and I will record my results on a separate device. I will keep it this way for all of the trials so that none are affected by other apps that are running.

Diagram:



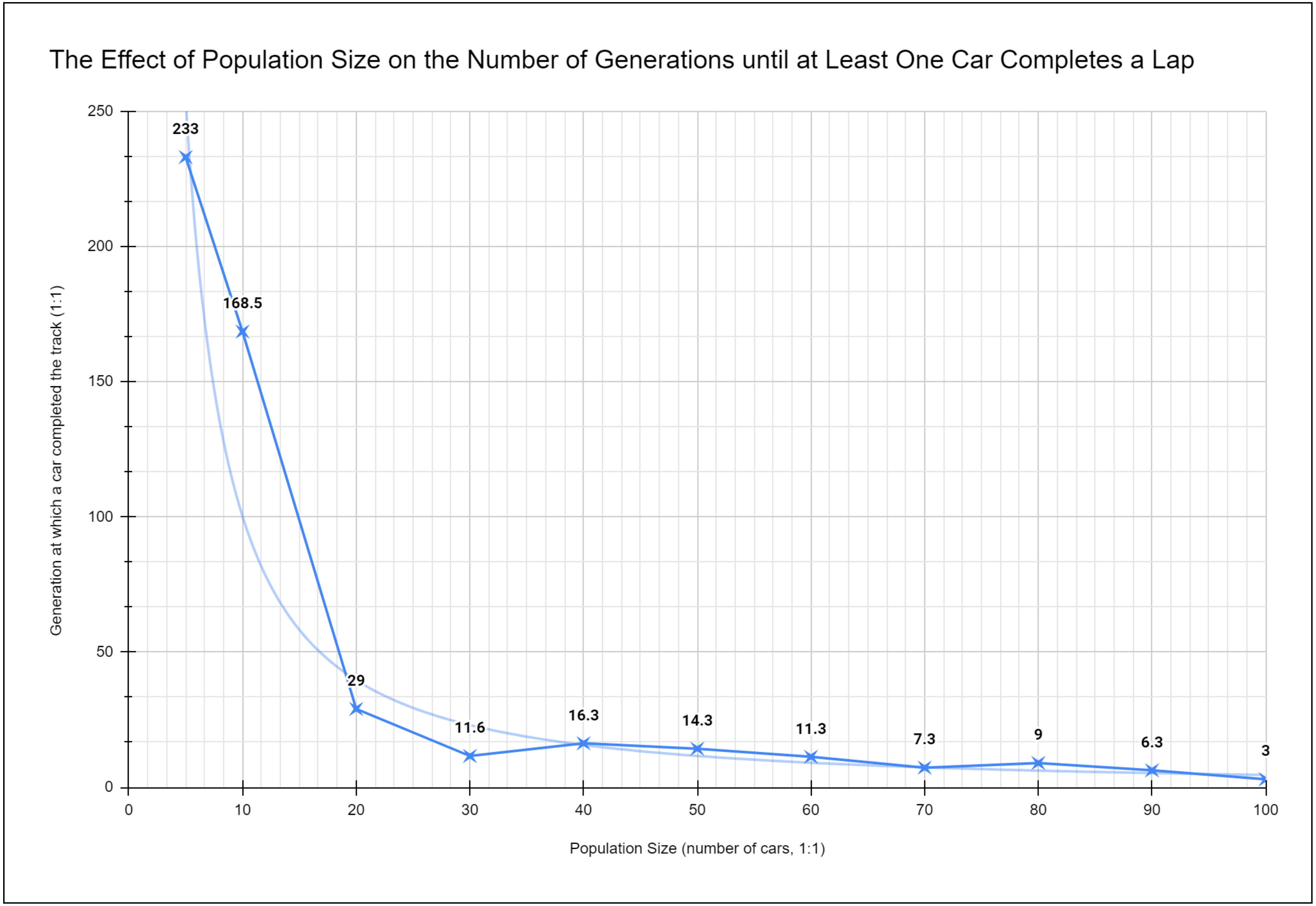
Method:

1. In the config.txt file, change the pop_size variable to 5.
2. Save the file.
3. Run the python file.
4. Watch for any cars displaying any of the behaviours listed in the control variables (going backwards, not moving).
    a. If any cars display these behaviours, shut down the program (Ctrl+Alt+F4) and go back to step 3.
5. Record the generation at which a car completes a full lap of the track.
6. Shut down the current program (Ctrl+Alt+F4).
7. Repeat steps 3-5 twice more.
8. Repeat steps 1-6 with a population size of 10, 20, 30, 40, 50, 60, 70, 80, 90 and 100.

Results:

The Effect of Population Size on the Number of Generations for At Least One Car to Complete a Lap

Population Size	Generation at which a car completed the track			
	Trial 1	Trial 2	Trial 3	Average
5	110	231	358	233.0
10	19	153	184	168.5
20	32	27	28	29
30	12	11	12	11.6
40	18	17	14	16.3
50	11	11	21	14.3
60	10	9	15	11.3
70	9	6	7	7.3
80	8	10	9	9.0
90	4	5	10	6.3
100	5	1	3	3.0



The Effect of Population Size on the Number of Generations until at Least One Car Completes a Lap
Generation at which a car completed the track (1:1)
Population Size (number of cars, 1:1)
250
200
150
100
50
0
0
10
20
30
40
50
60
70
80
90
100
233
168.5
29
11.6
16.3
14.3
11.3
7.3
9
6.3
3

Discussion:

I feel as though this experiment was conducted very well and very successfully. Through the controlling of a large number of variables, I was able to keep the investigation valid, and the method of measuring results was fairly accurate. The number of generations was counted by a variable in the code, removing the possibility for human error. The observation of a car completing a lap could have been mistaken, though due to the marking on the track I highly doubt that this is the case, increasing the accuracy of the experiment.

The analysis of the data reveals an overall decrease in the number of generations required to complete a lap, corresponding with the initial hypothesis that a higher population size decreases the number of generations before at least one car completes a lap. Notably, there is a sharp negative trend in the early population size values, but this trend smooths out and even slightly increases toward the higher population sizes, which is possibly due to increased computational lag when many cars are running simultaneously. There were a few outliers, however the most significant one resulted from trial 1 with 10 cars, which only took 19 generations for a car to complete a lap, as opposed to the other trials taking 153 and 184 respectively. To eliminate any further bias that this value may cause, it was omitted from the average calculations. Interestingly, a population size of 30 outperformed both 40 and 50, contradicting the overall trend. This potentially indicates a balance between the variation of mutations occurring and the likelihood of beneficial mutations being inherited in future generations. Occasionally, multiple cars completed the first lap simultaneously, suggesting that the beneficial mutations in previous generations were passed on to multiple of the future cars. Additionally, when the population size decreased, there was more variation between different trials. This could be because there were fewer mutations happening overall, and therefore even fewer beneficial ones. Sometimes, a beneficial mutation might occur earlier in one trial than in another, which would mean that the cars that learned a beneficial behaviour earlier would learn to complete the track sooner. The line of best fit that best suits the graph is a polynomial trendline, which is very steep for population sizes 5-20 and a lot gentler for the remaining population sizes. This particular line of best fit emphasises that the data is non-linear, and how finding a balance between the variation of mutations occurring and the likelihood of beneficial mutations being inherited is key to using the NEAT machine learning algorithm efficiently.

Possible improvements to this experiment include conducting more than 3 trials and having all of the variations of the independent variable running simultaneously. Running more than 3 trials would help lessen the observed variation in results between different trials, especially for the lower population sizes, potentially improving the reliability of the findings. Additionally, finding a way to run all variations simultaneously, such as assigning different-coloured cars to specific population size groups (e.g., group 1 with a population of 5, group 2 with 10, and so on) could allow the experiment to take less time. This simultaneous approach would also minimise the impact of potential lag or other background issues, as they would affect all of the cars uniformly. Implementing this approach might involve putting the current code into an overarching function or loop, with parameters like car colour and population size, and having it run once for each different population size. However, due to time constraints, I was unable to incorporate this idea into the code. Areas of further study could include looking more specifically at what population size affects, such as mutation rates. This would allow for further insight into how population size affects the number of generations for at least one car to complete a lap, rather than just identifying that it does. Introducing dynamic population sizing could also provide further information on this topic. This would involve the population sizes of different generations changing based on performance or other criteria, which could assist in balancing the number of mutations occurring with the chance of beneficial mutations being passed on. For example, the initial few generations could have a larger population to allow for more mutations to occur, and the later generations could have less so that beneficial mutations could have a higher chance of being passed on to the next generations. Overall, the experiment was successful, and the results can be analysed to determine that my initial hypothesis was correct, in that having a higher population decreases the number of generations for at least one car to complete a lap.

Bibliography: (Also including sources used for code commenting)

ChatGPT, 2023 (used to understand certain sections of the code)

Heidenreich, H. (2019) *Neat: An awesome approach to neuroevolution*. *Medium*. Available at https://towardsdatascience.com/neat-an-awesome-approach-to-neuroevolution-3eca5cc7930f

Kearney, W.T. (2016) *Using Genetic Algorithms to Evolve Artificial Neural Networks*, *Digital Commons @ Colby*. Available at https://digitalcommons.colby.edu/cgi/viewcontent.cgi?article=1836&context=honorstheses

NeuralNine. (2021) *Self-Driving AI Car Simulation in Python*. [Online]. Available at https://www.youtube.com/watch?v=Cy155O5R1Oo

Semper Zero. (2021) *Visualizing the NEAT Algorithm, (Series).* [Online]. Available at https://www.youtube.com/watch?v=j8oU0ksQ3Bc&t=2s

Shorten C. (2019) *Neuroevolution of Augmenting Topologies (NEAT)*. [Online]. Available at https://www.youtube.com/watch?v=b3D8jPmcw-g&t=635s

Smith, D. (2019) *Neat overview*, *NEAT-Python*. Available at https://neat-python.readthedocs.io/en/latest/neat_overview.html