Investigating the Impact of a Genetic Algorithm on Inter-Flock Dynamics in a Competitive Environment

Matthew James Hunter
BSc(Hons) Computer Games Technology
2019

School of Design and Informatics
University of Abertay Dundee

Contents

List of Figures

Acknowledgements

To Hayley McCullough who has somehow tolerated me during this.

I would like to thank Dr.Karen Meyer for her support as a supervisor.

Thankyou to Carol Clark who has been a great mentor and has given me increased confidence as the year has gone past.

Abstract

Context Flocking algorithms represent a kind of group behaviour that collectively improves the survival chances of individual members, and using genetic algorithms to improve a flock is an area already under exploration. However, not so looked into is the ability of a genetic algorithm to improve a flocks competitiveness versus a regular flocking algorithm and the changes in behaviours that may result.

Aim To develop an environment in which a flock that improves over generations via a genetic algorithm can compete over resources with a regularly implemented flock, using the expanded boids model. Exploring how the genetic algorithm affects the competitiveness of the flock in the environment in survival, potential convergence in evolutionary traits, and emergent situational behaviours.

Method A competitve environment with a limited resource pool not suitable to sustain two flock populations was developed. For this environment a flocking algorithm and genetic algorithm were developed.

Results

Conclusion

1 Introduction

1.1 General Introduction and Background Information

Flocking is an emergent property of social behaviours, it is the common movement of individuals guided by both social and environmental influences, and is one in which many social organisms engage. These flocks of organisms can be found interacting with other flocks in nature, and the way they interact is as varied as it is interesting.

Producing flocking behavior that recreates those found in nature is an endeavor already undertaken and in constant update. The field, taking off in 1987 with Craig Reynolds' influential paper (Reynolds, 1987), shows how realistic flocking behaviour can already be achieved by applying 3 simple rules to each boid ('boid' as coined in the prior paper, this is essentially an agent) in regard to its neighbours: Cohesion, movement toward the average position; Alignment, movement toward the average direction; and Separation, movement to avoid collision.

Since then, the original flocking algorithm has been extended in various ways, with communication techniques, mathematical models for how leadership arises in the flock, as well as models for how consensus is made in a flock. These expansions allow for more complex behaviours and reactions to the enviornment and surroundings.

Learning behaviours have also been added. The behaviour and strategies flocks produce are patterns. This means if a flock can learn and understand those patterns it has a significant advantage over that other flock – an interesting example would be a group of honey hunters and smoke; bees flee the nest if they think there is a fire, the first warning sign to this is smoke, as a group (or flock) they take advantage of this by releasing smoke into the hive. The bees evacuate the nest; they get honey.

This dissemination of new knowledge, either through behaviour or communication is interesting because it can increase the complexity of the reactions the flock has to a given situation. This added complexity may lead to new behaviours that may not have been easily predicted or thought of as something a flock could produce.

1.2 Aim, Objectives and Research Question

Aim To investigate if a flock improved by a genetic algorithm can outcompete a regular flock in a competitive environment, and to identify what changes the genetic algorithm produces, convergences and the kinds of behaviours produced over time.

Objectives

- To research and evaluate AI techniques, studying their relevance and potential for further development in applying them to flocking algorithms, with particular focus on artificial life techniques.

- To produce an application that models flocking behaviour, and allows the observation and comparison of AI flocking strategies to regular flocking algorithms. This will be developed using the prior identified AI techniques most likely to produce viable flocking behaviours.
- To analyse the effectiveness of strategies that the AI come up with in their interactions with other flocks, comparing and contrasting that to the behavior of standard flocking algorithms.

Research Question What impact can a genetic algorithm have on the dynamic interaction of flocks with each other in comparison to that of regular flocking techniques?

2 Literature Review

This chapter will identify the gap in research that forms the basis for the research question, by reviewing existing work on flocking and genetic algorithms, and focusing down from the wider fields of research it will frame the context of this research paper.

2.1 Flocking in Nature - Swarm Behaviour

Flocking algorithms draw a lot of their inspiration from the behaviour of flocks in the natural world (Flake, 1998). As there are many examples of this behaviour in organisms across the planet, there is a lot of information and insight that can be gleaned on how to design these algorithms. Flocks are a kind of swarm behaviour, and looking at the broader set of behaviors can point to the interesting ways a flock could behave under certain conditions.

Decision Making The way decisions are made in a flock emerges is varied. The two main ways are via consensus and leadership. An interesting look at this can be found in the behaviour of pigeons. A study conducted into the behaviour of these birds in a flock (Jorge and Marques, 2012) found that leadership initially emerged from younger pigeons in the flock, but as the flight went on older members of the flock led the group, the paper then goes on to discuss how social versus personal information affects the behaviour of the flock. What this displays, is how the extra experience of older members of the flock is taken advantage of in determining leadership and therefore the actions they take as a flock, in this way they build consensus on their leadership through the choices individual flock members make in their group. This is further confirmed in *'Misinformed leaders lose influence over pigeon flocks'* Watts et al. (2016). What this displays is the interaction between consensus and leadership in making decisions, and demonstrates that they can both be present in the process.

Eusocial Behaviour While not specifically flocking as it pertains to biology, this falls under the bracket of swarm behaviour and has interesting relationships which can inform us on how to expand flocking algorithms in beneficial ways. For example, communication can occur in a variety of ways; ants use pheromones to find shortest paths (Dorigo and Gambardella, 1997); bees use a waggle dance to inform others in the hive of food sources and potential new nest sites (Al Toufailia et al., 2013), and

Dynamic Adaptation to Environmental Pressures

Learning and Curiosity

Flocking and Emergence

The Advantages of Flocking to Species

Why this is relevant This is relevant as it inspires the work done in flocking algorithms and the work of similar fields, which has influence on the design of said algorithms.

2.2 Flocking Algorithm

Flocking algorithms draw inspiration from the natural world, however the design of these algorithms involves mathematical approximations of the behaviours involved, the interactions that take place and the systems overall.

Reynolds Boids In the often cited paper when it comes to discussions of flocks: *'Flocks, herds and schools: A distributed behavioral model'* Reynolds (1987), we see that we have the three main forces that make up a basic flock: Separation, Cohesion and Alignment. These are approximative forces that represent the aggregate motion of a collection of boids (representing flock members, which in turn can represent different species in nature).

Expanded Boids There are many variations of the expanded boids model [insert citations here] dependent upon what is necessary for the research conducted. The expanded model presented by Kawabayashi and Chen (2008) is very useful as their implementation is very clear and displays useful expansions to the model for this research. It adds functionality for running away from predators, attraction to food sources, relations as towards other neutral flocks, the ability to search its surroundings, boundary conditions, and obstacle avoidance. All of which are useful in designing more natural flocking behaviour.

Decision Making In parallel to the way decisions are made in natural flocks, flocking algorithms can also make decisions. An interesting way of producing leadership in an artificial flock can be found in the paper textit'Autonomous Boids' Hartman and Benes (2006). Here they propose the use of an 'Eccentricity' variable to determine leadership in the group in order to make decisions based on the boids proximity to the front of the flock, the closer it is, the higher a chance of leading the flock. This mimics the way that some species of bird, such as starlings, decide on leadership within the group. This leadership can then be used to influence decisions by the rest of the flock.

Learning and Curiosity

2.3 The Interaction of Flocking Algorithms

In the paper *'Simulating Species Interactions and Complex Emergence in Multiple Flocks of Boids with GPUs'* Husselmann and Hawick (2011), multiple flocks,

each a different type of boid, are run in a closed environment to see what aspects of species interaction could be reproduced. The

2.4 Genetic Algorithm

The literature on genetic algorithms also matters as this is what will be used to expand upon the capabilities of the flock. There is

2.5 The Potential Effect of AI on Flocking Algorithms

A paper discussing the use of reinforcement learning in a multi-agent system by Jaderberg et al. (2018) displays the potential of AI applied to a group of agents.

2.6 The effect of Genetic Algorithms on Boids

Genetic Algorithms used to improve the boids model is nothing new and provides useful information into how to go about this research. For example in *'Genetic Algorithms for Optimization of Boids Model'* Chen et al. (2006)

2.7 The Interaction of an AI Flock with another Flock

The interaction of an artificially intelligent flock with other flocks is where the research becomes thinner and where the author feels based upon their own background research there is a room for expansion.

3 Methodology

An application was produced to test whether a flock improved by a genetic algorithm can outcompete a flock using a regular expanded boids algorithm in a competitive environment. This application used the Games Education Framework (GEF) [reference] as the framework, which was chosen based on its lightweight overhead, its portability and recommendations from several university lecturers. It was developed using Visual Studio 2017 in C++, chosen due to the ease of working with them and the flexibility afforded by both tools.

GitHub Desktop was used as the tool for source control, ensuring risks to the project were kept to a minimum. Further to this, a GitFlow tool was developed in order to stem the risk of pushing and pulling work from incorrect branches, and prevent any conflicts when committing and pushing new work. The tool essentially acted as a smart checklist; starting a programming session with meant everything was set up correctly before opening individual programs, which happened automatically as the correct items were ticked off.

3.1 Test Environment Design

The application simulated an environment in which two flocks interact (one of which is improved via a genetic algorithm) within a limited space of scarce resources, with the aim of finding out if a flocking algorithm improved by a genetic algorithm can out-compete one without.

To answer the research question required removing as many base assumptions as possible. To do this, three core decisions were made: The environment would be limited so as to keep the flocks in competition with each other, ensuring the flocks would interact; there would be a limited amount of resources in the environment: less than that required to support two full populations of flocks - this would drive the flocks to compete over resources spawned in the environment; finally there would be only two flocks: one improved by a genetic algorithm over simulated generations and one produced through a tested expanded flocking algorithm design. This ensured a competitive environment in which the effects of decisions made by genetic algorithm could be singled out, and the data producing more accurate results. Fig.1 displays a screenshot of this environment in action: the red entities representing the regular flock; the blue entities representing the genetic algorithm modified flock; the green entities representing resources in the scene and the white area displaying the space in which the boids can move and interact.

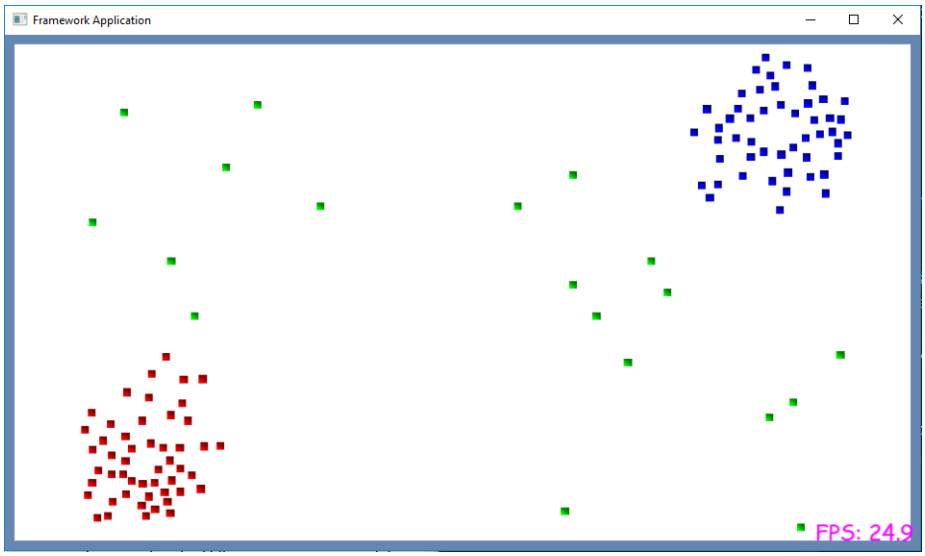

Figure 1: A screenshot of the test environment in action

3.2 Flocking Algorithm Design

The original flocking algorithm was designed primarily by adapting the example from Shiffman (n.d.). This was a very clear representation of how to design a basic flocking algorithm from a code perspective. There were also clear areas for improving the efficiency of the algorithm presented. The most effective way was in combining the calculations for Alignment, Separation and Cohesion. This was to ensure each boid was not checking against all others for each force calculation, and only needed to check against all other boids once instead of three times; doing that was simple and reduced the requirement for N-Body problem solutions like the Barnes-Hut algorithm later on. The other main way was combining the calculations required for each force as they used shared variables, so calculating them once instead of multiple times for each force also increased efficiency.

This worked adequately for the original three forces based on Reynolds (1987). However, issues came up when expanding the application to include extra forces. These were heightened by the fact that, in the aforementioned implementation, it wasn't strictly clear how the forces would graph out due to the lack of separation between direction and weighting. This caused issues to accumulate and it was clear a change in the flocking algorithm design was necessary to expand and edit the program appropriately.

The boids' forces as they are in the final program were modelled primarily off Kawabayashi and Chen (2008), which has a very clear mathematical representation of each force. The version of the expanded boids algorithm found in this paper was modified by optimising the algorithm for the environment in

which it was placed. The forces each boid experienced were: Cohesion, Alignment, Separation, Food Attraction and Flock Avoidance. Each of these forces were the result of multiplication of a unit vector and their respective weight, mathematically represented in Eq.1

$$\boldsymbol{ForceVector} = \boldsymbol{v} \cdot \boldsymbol{w} \tag{1}$$

Where $\boldsymbol{v}$ is the unit vector which describes the direction of the force, and $\boldsymbol{w}$ is the weight that magnifies the force dependent on its relation to its environment.

The relationship of the weight to the unit vector will be detailed in relation to boid forces below. Separating direction vector and weight meant it was far easier to mathematically model and graph out each force to predict their behaviour; this reduced in the amount of work necessary to produce a functioning flocking algorithm. The resulting accelerative force then was the sum of these forces. That sum was then used in the Semi-Implicit Euler method to produce the updated position of each boid, every frame.

3.2.1 Cohesion

The first of the boid forces, Cohesion pulls the boid in the direction of flock members, in this case towards its local flock centre. The boids' local flock centre is determined by its neighbours in communicable range and is the average position of those flock members. The further the distance away from its neighbours, the stronger the force. Below is Eq.2 describing the direction vector and weighting:

$$\begin{aligned} \hat{\boldsymbol{v}}_{\boldsymbol{coh}} &= \frac{LFCVector}{|LFCVector|} \\ \boldsymbol{w}_{\boldsymbol{coh}} &= \frac{(|LFCPos - BoidPos|)^2}{30 \cdot FlockSize} \end{aligned} \tag{2}$$

Where $LFCVector$ represents the vector from the boid to the local flock centre, $LFCPos$ is the position of the local flock centre, $BoidPos$ is the position of the boid and $FlockSize$ is the number of flock members as a whole.

This can be seen visually represented in Fig.2. It displays the cohesion force increasing as a boid moved away from the local flock centre.

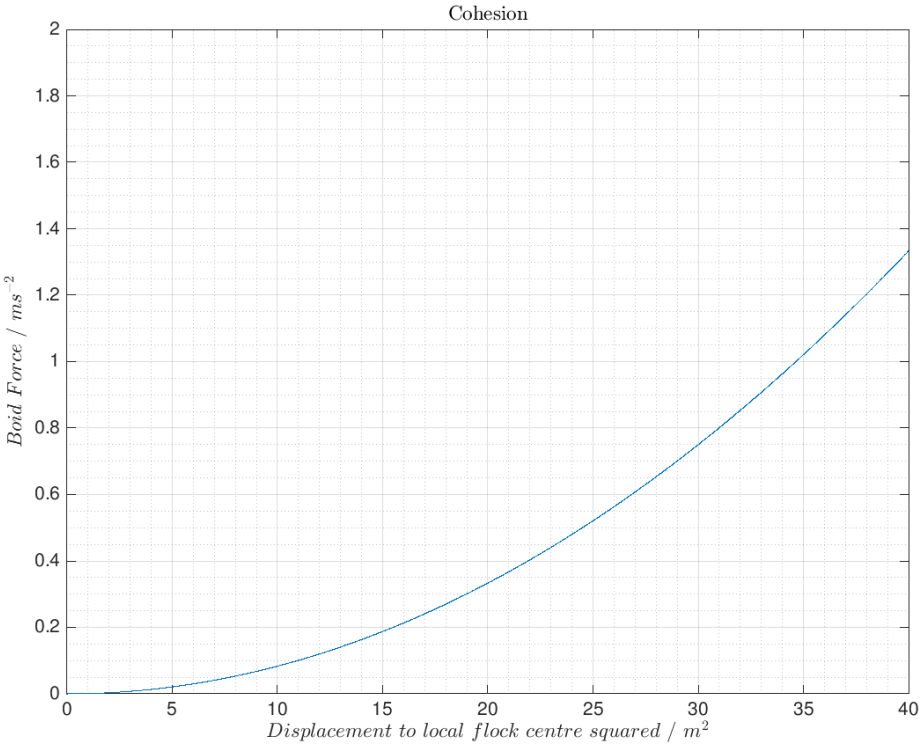

Figure 2: A Graph Displaying the Cohesion Boid Force

3.2.2 Alignment

Alignment is a vector which accelerates the boid in the average direction of its neighbouring boids. This produces the common direction of the flock as the boids individually move around. By taking the average velocity of neighbouring boids as a direction vector and multiplying it one over the distance between the local flock centre and the boid itself, a suitable alignment vector is produced. Below is Eq.3 describing the two components of the alignment vector:

$$\begin{aligned} \hat{\boldsymbol{v}}_{\boldsymbol{ali}} &= \frac{LFVelVector}{|LFVelVector|} \\ \boldsymbol{w}_{\boldsymbol{ali}} &= \frac{1}{10 \cdot |(LFCPos - BoidPos)|} \end{aligned} \tag{3}$$

Where $LFVelVector$ is the average velocity of the local flock members, and $LFCPos$ and $BoidPos$ are the same as in Eq.2.

This can be seen visually represented in Fig.3. It displays the alignment force increasing as a boid moved closer to the local flock centre.

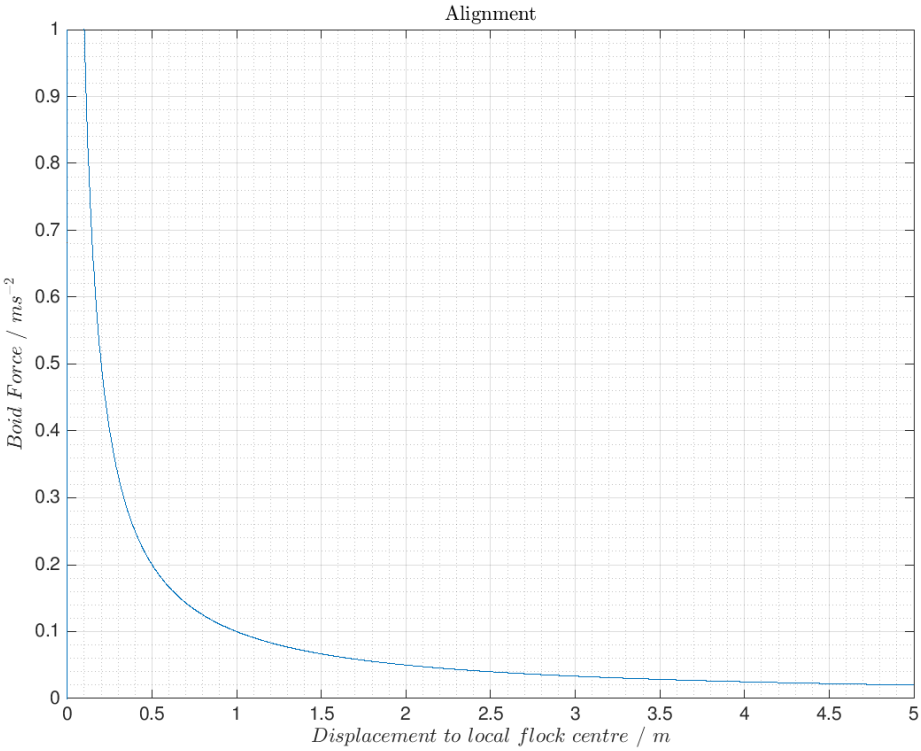

Figure 3: A Graph Displaying the Alignment Boid Force

3.2.3 Separation

Separation is the force that stops the boids from getting too close together. By maintaining enough space between flock members, it grants maneuverability where there would otherwise be collisions (which slow movement down); these could be from obstacles, other moving entities or other boids in the flock. The unit vector that represents the direction for the force to be applied is calculated by taking the negative of the vector to the nearest neighbouring boid; multiplying that by the weighted multiple of the number of neighbours divided by the distance to the closest neighbour produces the separation vector described below in Eq.4:

$$\begin{aligned} \hat{\boldsymbol{v}}_{\boldsymbol{sep}} &= -\frac{ClosestNeighbourVector}{|ClosestNeighbourVector|} \\ \boldsymbol{w}_{\boldsymbol{sep}} &= 0.025 \cdot \left(\frac{NeighbourCount}{|ClosestNeighbourVector|}\right)^2 \end{aligned} \tag{4}$$

Where $ClosestNeighbourVector$ is the vector from the boid to the closest neighbour, and $NeighbourCount$ is the number of neighbouring boids.

This can be seen visually represented in Fig.4. It displays the separation

force increasing as the displacement between the nearest flock member and itself decreased. This separation force increased for each neighbouring boid.

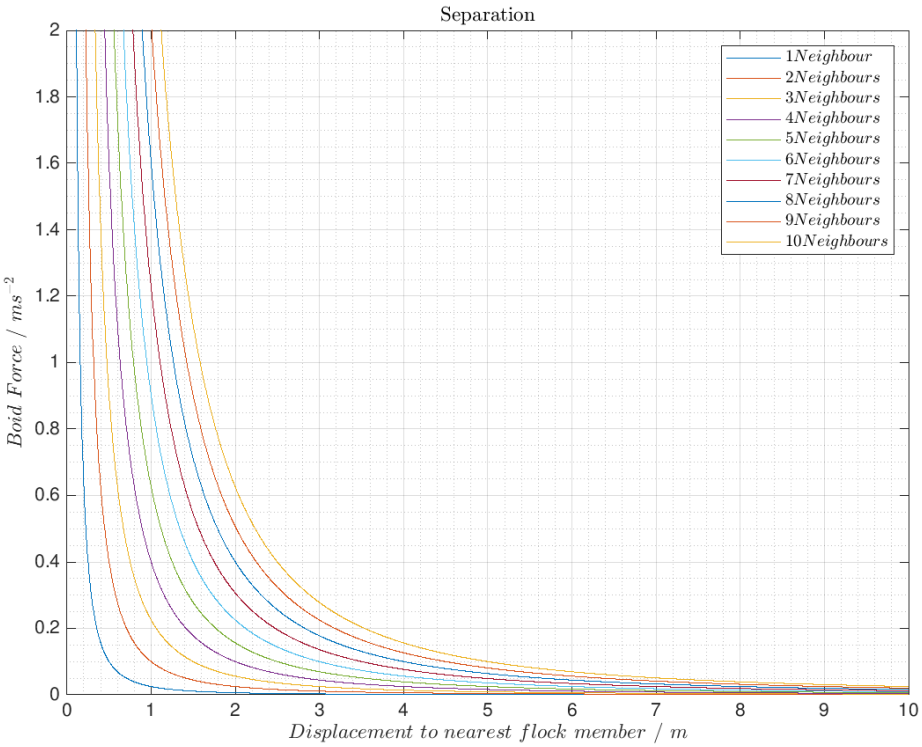

Figure 4: A Graph Displaying the Separation Boid Force

3.2.4 Food Attraction

This provides an accelerative force toward the nearest resource in the environment. This attraction to nearby food sources is the basis for the boids' survival in the scene. The unit vector to the closest resource gives the direction, and it is the weight which gives its relationship to food in terms of distance away. The equations used are described in Eq.5:

$$\begin{aligned} \hat{\boldsymbol{v}}_{\boldsymbol{fda}} &= \frac{ClosestResourceVector}{|ClosestResourceVector|} \\ \boldsymbol{w}_{\boldsymbol{fda}} &= 0.0025 \cdot |ClosestResource|^2 + \frac{36}{|ClosestResource|^2} \end{aligned} \tag{5}$$

Where $ClosestResourceVector$ is the vector pointing from the void to the closest resource to it, and $|ClosestResource|$ is the distance to the closest resource.

This can be seen visually represented in Fig.5. It displays relationship the food attraction force has to the displacement squared of the food resource to

the boid. It represents two increases in the food attraction force and a minima inbetween, where the food attraction force decreases as the boid gets near to it, but then switches to an increase past a certain point.

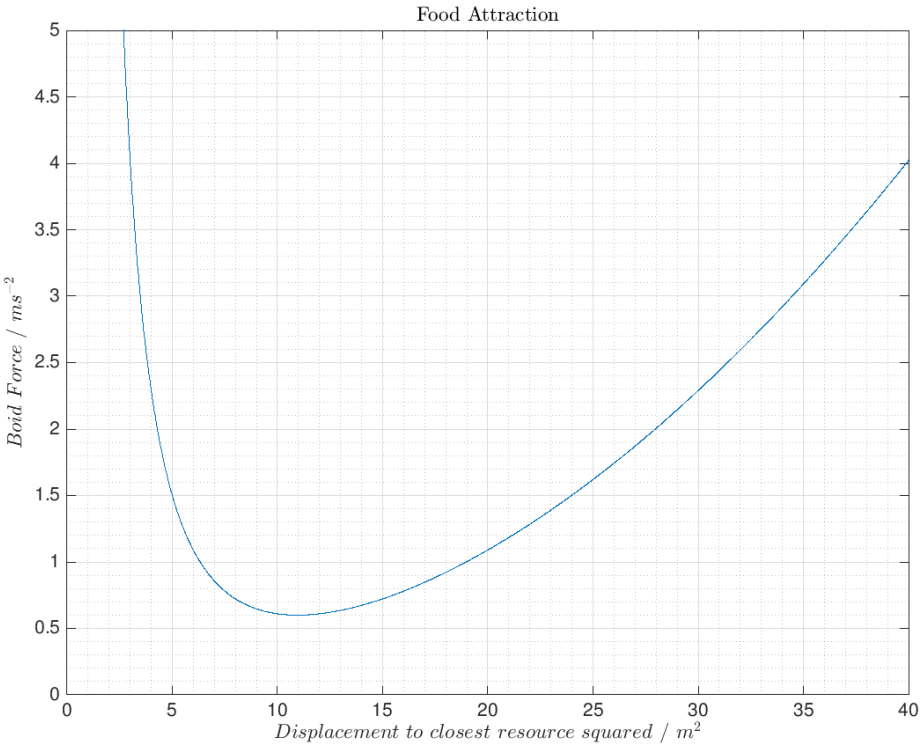

Figure 5: A Graph Displaying the Food Attraction Boid Force

3.2.5 Flock Avoidance

This is a vector that produces a repelling force away from another flock, and increases as the boid gets closer to the other flock; this is what stops the boids from ignoring each other and keeps them in a competitive environment as they will both be competing for those same resources. The mathematical relationship of the two factors produced can be seen in Eq.6:

$$\begin{aligned} \hat{\boldsymbol{v}}_{\boldsymbol{fla}} &= -\frac{OtherFlockVector}{|OtherFlockVector|} \\ \boldsymbol{w}_{\boldsymbol{fla}} &= \frac{300}{|AvgOtherFlockPos|} \end{aligned} \tag{6}$$

Where $OtherFlockVector$ is the vector pointing from the boid to the average position of another flock within interactable range, and $AvgOtherFlockPos$ is the vector to the average position of another flock within interactable range.

This can be seen visually represented in Fig.6. It displays the flock avoidance force increasing as a boid moved closer to the local flock centre.

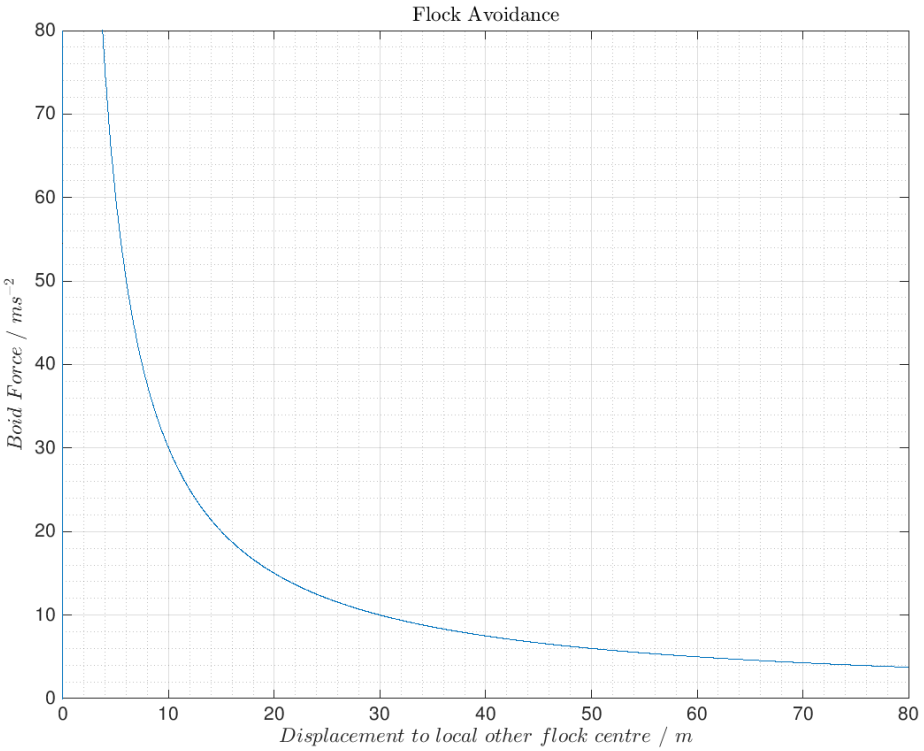

Figure 6: A Graph Displaying the Flock Avoidance Boid Force

3.3 Genetic Algorithm Design

The functionality for the genetic algorithm (GA) was spread over two classes - genetic_algorithm and DNA - where the genetic_algorithm class interacted with the flock and boid classes, and DNA interacted with the genetic_algorithm and boid classes. The GA had a standard approach to its design which can be seen in Fig.7.

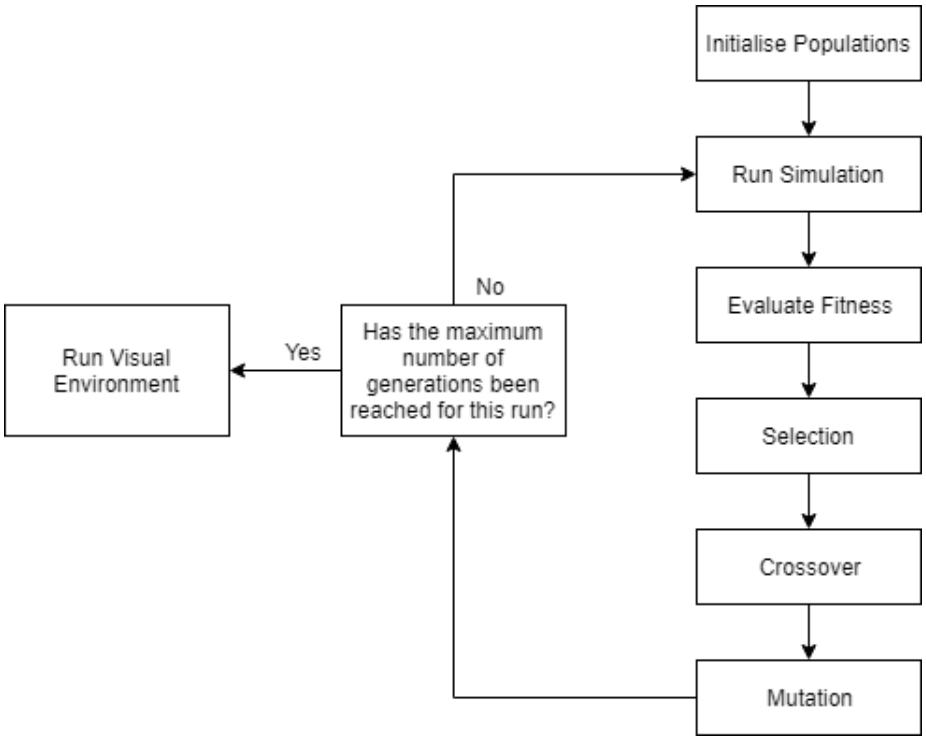

Figure 7: Flowchart displaying the process the GA takes to improve over time

Five populations were generated each time the algorithm was initialised; it could be set up to be either random or heuristic, where each population had a random set of data generated for its alleles, and the same but where one population received the heuristic used in the regular flocking algorithm respectively. These populations of boids then made up the genetically enabled flock.

Each population produced contained chromosomes unique to that population, which is represented by an array in the DNA class. Each GA boid had its own DNA object which is modified by the genetic_algorithm class. Collectively they made up their respective genotype. Each gene in a chromosome is used in the weight calculations of its respective boid; this can be seen represented below

in Eq.7:

$$
\begin{aligned}
\boldsymbol{w_{coh}} &= \frac{(chromo[0] \cdot |LFCPos - BoidPos|)^2}{chromo[1] \cdot FlockSize} \\
\boldsymbol{w_{ali}} &= \frac{chromo[2]}{chromo[3] \cdot |(LFCPos - BoidPos)|} \\
\boldsymbol{w_{sep}} &= chromo[4] \cdot \Big(\frac{chromo[5] \cdot NeighbourCount}{chromo[6] \cdot |ClosestNeighbourVector|}\Big)^2 \\
\boldsymbol{w_{fda}} &= chromo[7] \cdot |ClosestResource|^2 + \frac{chromo[8]}{chromo[9] \cdot |ClosestResource|^2} \\
\boldsymbol{w_{fla}} &= \frac{chromo[10]}{chromo[11] \cdot |AvgOtherFlockPos|}
\end{aligned}
\tag{7}
$$

Where $chromo[x]$ represents each gene in the chromosome for that boid, changes of the allele of each gene affect the weights in the scene. These weight calculations made up the phenotype representation in the environment space, and determined the behaviour of a boid.

3.3.1 GA Methods

Evaluation At the end of each simulation, each boid was evaluated against the fitness function, which can be seen in Eq.8. The evaluation function then sorted the boids into population types for the selection process.

$$
FitnessFunction = BoidHealth + \boldsymbol{w_1} \cdot FlockHealth - \boldsymbol{w_2} \cdot OtherFlockHealth \tag{8}
$$

Where $BoidHealth$ is the health of the boid, $FlockHealth$ is the combined health values of all boids in the flock, and $OtherFlockHealth$ is the combined health of all boids in the competing flock.

This fitness function in Eq.8 was designed to take into account the three important factors to a boid in a competitive flock: accounting for its own health was to represent a survival instinct, adding the total flock health to the equation meant improvements to itself over time should be in benefit the flock as a whole, and taking away the weighted total health of the other flock meant the boid had a focus on improving by decreasing the health of the other flock. This fitness function was intended to produce behaviour in the boids that outcompeted the regular flocking algorithm in the environment.

Selection The application contained two different kinds of selection algorithm: Roulette Wheel Selection (RWS) and Stochastic Universal Sampling (SUS), both used in answering the research question. In both selection algorithms, increased probabilities of being chosen were given to those populations with better overall fitness. These methods were chosen as they provided selection pressure toward fitter individuals, while also not prematurely coming to a solution which

may have been more likely to be a local maxima.

Crossover Crossover of genetic data between parents was done as a single-point crossover that was randomly selected along the genotype, once selected the geneotype halves then got swapped. This was selected due to it being one of the more efficient types of crossover involving the least calculations.

Mutation Mutation occured via random increases or decreases of an allele in a randomly selected gene in the genotype, this was intended to be slow enough to not skip over solutions and fast enough to come to solutions quicker.

3.3.2 Generations

Part of running the GA was simulating the environment in sped-up runs for the genetic algorithm to improve itself over successive generations. This was done by cutting off unnecessary calculations, that meant all generations occurred in the update function in a loop until the maximum number of generations had been reached. This cut out the requirements for rendering or indeed any calculations not related to the simulation of boid movement and running the genetic algorithm.

The calculations for the simulation physics were also cut down to speed up simulation time in each generational loop. By increasing the time per physics calculation you effectively speed up the time passed each simulation. For example: If one second of simulation time happens in one frame, then if the application runs at sixty frames per second, you can get sixty second of simulation time per real time second, this is displayed in Eq.9.

$$\boldsymbol{SimTime}(s^{-1}) = SimTimeStep \cdot FramesPerSecond \tag{9}$$

Where $SimTime$ is the amount of time passed in simulation per actual second, and $SimTimeStep$ is the difference in time for each calcualtion

Based on this sped up time, each generation had a simulation time limit before moving onto the next one. This had to be balanced as it needed to be a long enough time for real data to come through to get an accurate estimation of fitness and short enough so that the genetic algorithms improvement didn't take too long. The amount of time given per generation will be discussed in the results section as a factor that is controlled for each test due to this.

3.4 Testing Criteria and Metrics

The testing criteria has been designed with answering the research question in mind.

3.4.1 Controlled Factors

In a simulations with so many variables, there were many to consider controlling for to identify trends that might otherwise be obscured. In this section will be

factors identified as necessary to control for when running the simulation and why. Factors that need to be controlled have been mentioned alongside the results produced where relevant.

Initialising populations was controlled due to two methods being developed in the application, having this controlled meant that it could be made sure there was no effect from changing the population initialisation method. As there were also two different selection menthods developed, controlling for this was important as it was necessary to know in order to make sure runs of the simulation were not accidentally mixed and matched when relevant to producing results, as this could affect the quality of discussion around them and may point toward false conclusions. The selection was always the same during a run of the simulation and so was a factor to consider when producing the data sets. However, this does not need to be controlled for when looking for potential points of convergence in values for each of the boids alleles, as this points toward potential maximas of the solution and is not specific to a selection method.

The probability of mutation was a control factor, keeping it at a fixed value meant the tests could be compared in a temporal manner across generations, especially considering the potential for convergence of data over time. The amount of boids placed in the scene each generation being kept constant meant that the amount of competition for resources was constant across generations and simulation runs. This was similar to the number of resources in the scene, this amount of resources meant more competing with the other flock over food resources, increasing it or decreasing it in different runs could have had an effect on the strategies the flock developed.

3.4.2 Data Analysis

Allele The frequency of each allele in a gene was used to identify top alleles, and identify if there was convergence of values across simulations. If there was any convergence then that has the potential to be pointing toward maxima in the solution that can point toward the global optimum.

Genotype Analysing the genotype values will give some key results

Phenotype How did the genotypical representations identified as important actually have an affect on the boid weights, and so therefore its behaviour in the environment?

GA Flock Strategies We will identify observations made about the

4 Results

4.1 Testing the Effect of Sampling Method

The aim of this test was to identify the effects the sampling method may have had on the genetic algorithm. There are two implementations of sampling method in the program: Roulette Wheel (RW), and Stochastic Universal Sampling (SUS). Two tests were conducted, using each sampling method respectively, the below factors were standardised between the tests:

- Number of Generations: 20000
- Population Generation: Randomised Method
- Mutation Probability: 1%
- Time Step: 1.5 Simulation Seconds per frame

4.1.1 Roulette Wheel Data

4.1.2 Stochastic Universal Sampling Data

4.1.3 Data Comparison

4.2 Testing the Effect of Mutation Probability

This was a test to identify the affect mutation probability had on the genetic makeup of the flock and to identify the affect it may have on the effectiveness of the genetic algorithm as a whole. This required standardising the tests in each run of the GA, these were:

- Number of Generations: 5000
- Population Generation: Randomised Method
- Selection Type: Stochastic Universal Sampling
- Time Step: 1.5 Simulation Seconds per frame

Keeping these factors controlled across all tests, 10 tests were then produced with different mutation probabilities: 0.5%, 1%, 2%, 3%, 4%, 5%, 10%, 15%, 20%, and 25%.

4.2.1 The Effect on Fitness

Below is the data collected regarding the fitness value of the boids over each generational improvement. This data will shine a light onto the effect the mutation rate has on the boids fitness.

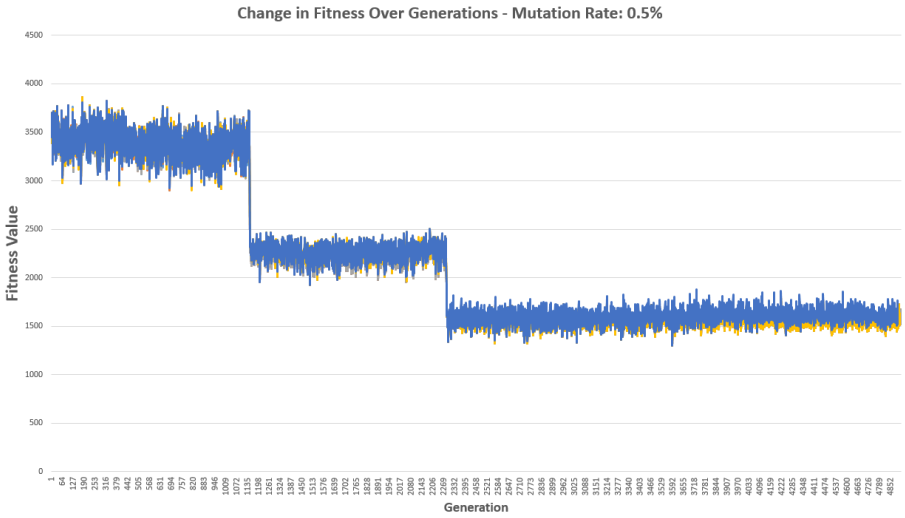

Figure 8: A graph displaying the fitness each generation with a mutation rate of 0.5%

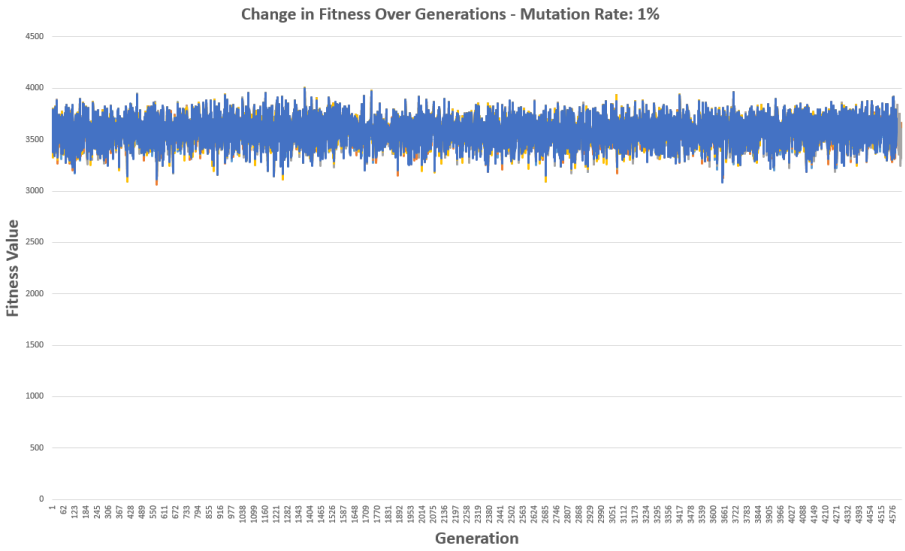

Figure 9: A graph displaying the fitness each generation with a mutation rate of 1%

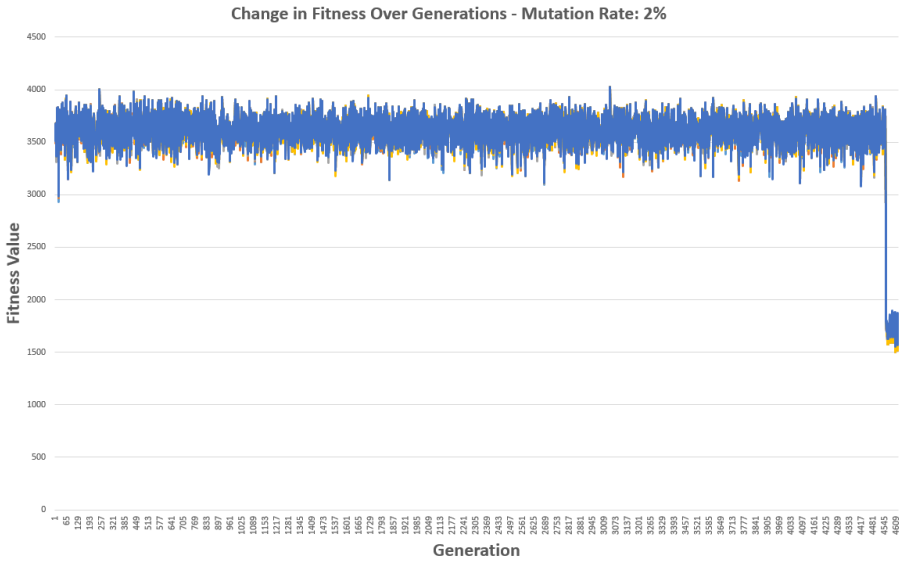

Figure 10: A graph displaying the fitness each generation with a mutation rate of 2%

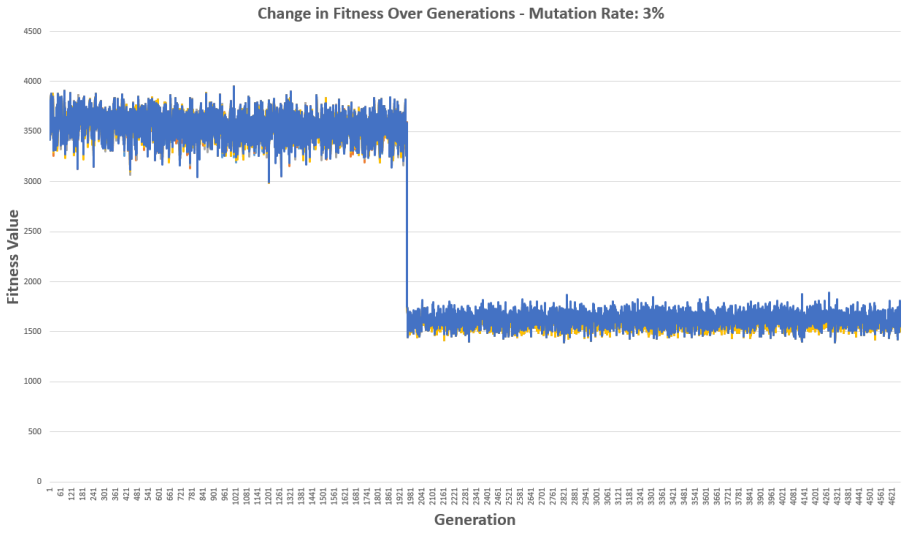

Figure 11: A graph displaying the fitness each generation with a mutation rate of 3%

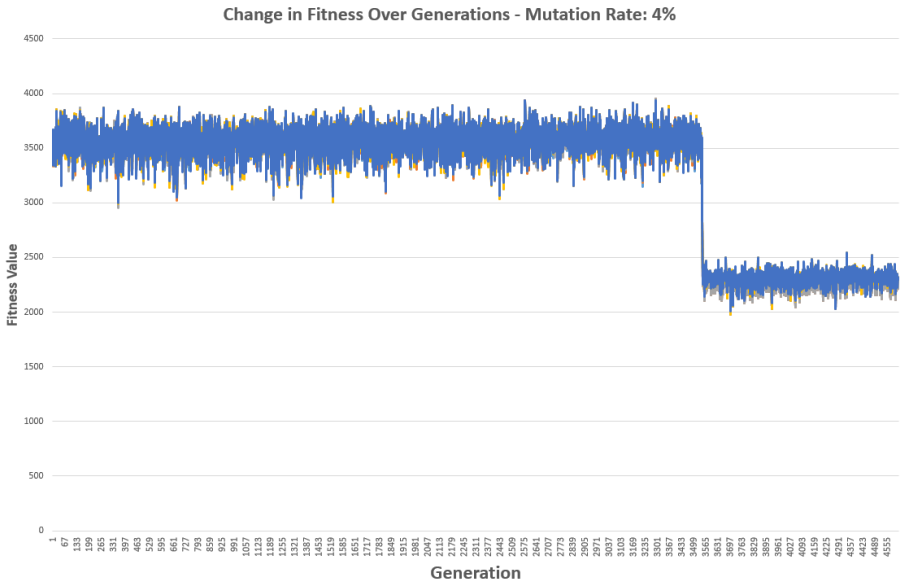

Figure 12: A graph displaying the fitness each generation with a mutation rate of 4%

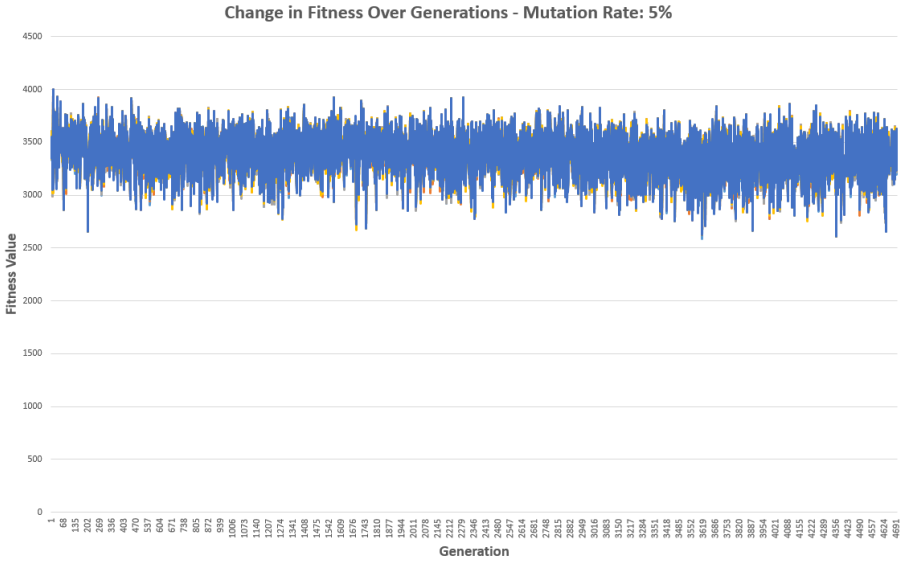

Figure 13: A graph displaying the fitness each generation with a mutation rate of 5%

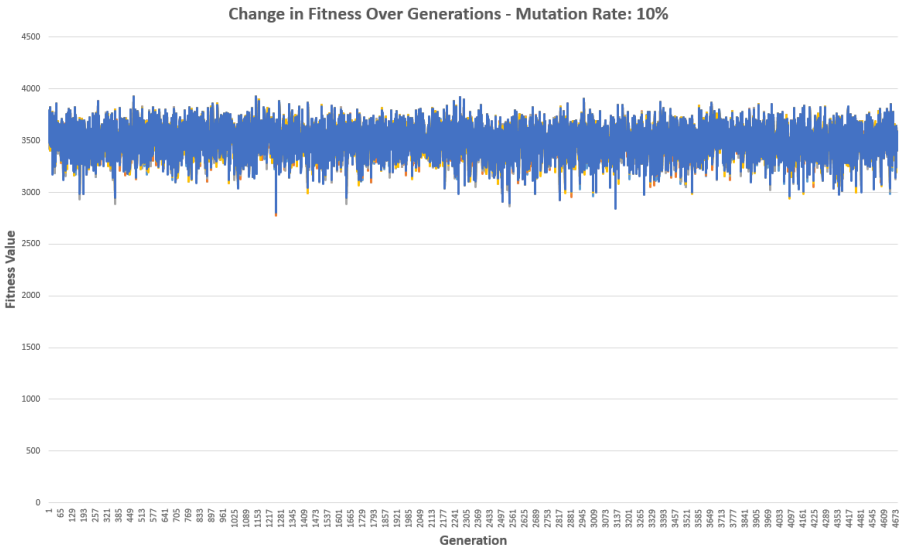

Figure 14: A graph displaying the fitness each generation with a mutation rate of 10%

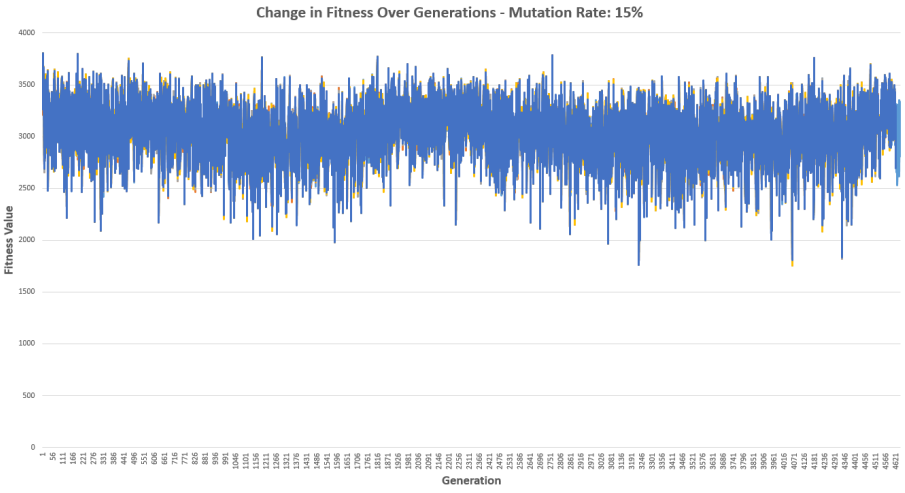

Figure 15: A graph displaying the fitness each generation with a mutation rate of 15%

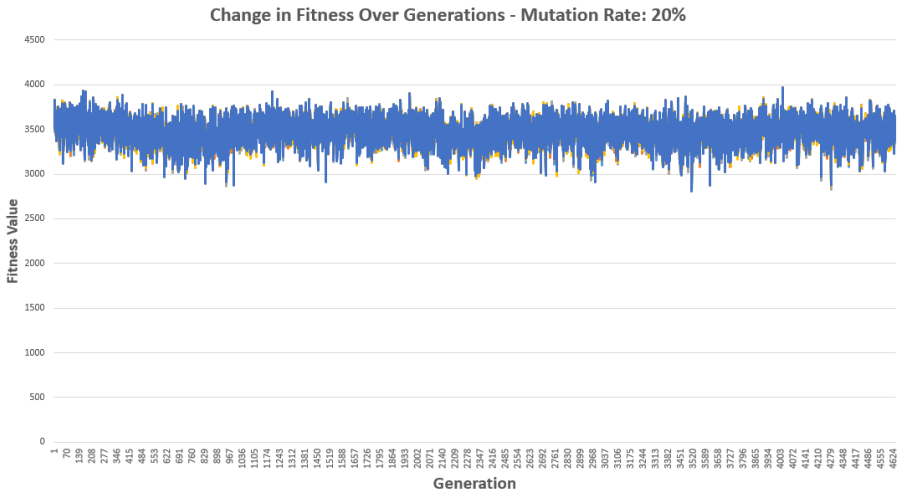

Figure 16: A graph displaying the fitness each generation with a mutation rate of 20%

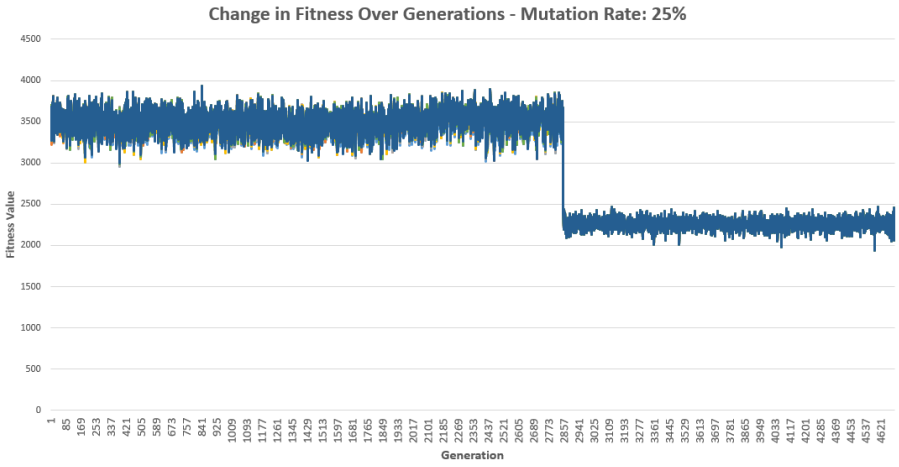

Figure 17: A graph displaying the fitness each generation with a mutation rate of 25%

4.3 Allele Convergence

Using the data collected from the previous tests, the aim was to analyse the frequency of different alleles to identify the most frequently produced values in the simulation as this may point toward optimal solutions for the boid algorithm.

4.4 Genotype Convergence

4.5 Phenotype Effects of Common Genotypes

4.6 Strategies

By visually analysing the flocks in action

5 Discussion

This chapter will identify the potential significance of results of the tests carried out, by highlighting key trends, producing observations, and

5.1 Sampling Method

References

Al Toufailia, H., Couvillon, M. J., Ratnieks, F. L. W. and Grüter, C. (2013), 'Honey bee waggle dance communication: signal meaning and signal noise affect dance follower behaviour', *Behavioral Ecology and Sociobiology* **67**(4), 549–556.
URL: *https://doi.org/10.1007/s00265-012-1474-5*

Chen, Y.-W., Kobayashi, K., Huang, X. and Nakao, Z. (2006), Genetic algorithms for optimization of boids model, Vol. 4252, Springer Verlag, pp. 55–62.

Dorigo, M. and Gambardella, L. M. (1997), 'Ant colonies for the travelling salesman problem', *Biosystems* **43**(2), 73 – 81.
URL: *http://www.sciencedirect.com/science/article/pii/S0303264797017085*

Flake, G. (1998), *The Computational Beauty of Nature: Computer Explorations of Fractals, Chaos, Complex Systems, and Adaptation*, A Bradford book, Cambridge, Massachusetts.
URL: *https://books.google.co.uk/books?id=0aUhuv7fjxMC*

Hartman, C. and Benes, B. (2006), 'Autonomous boids', *Computer Animation And Virtual Worlds* **17**(3-4), 199–206.

Husselmann, A. and Hawick, K. (2011), Simulating species interactions and complex emergence in multiple flocks of boids with gpus, *in* 'Proc. IASTED International Conference on Parallel and Distributed', IASTED, pp. 100–107.

Jaderberg, M., Czarnecki, W. M., Dunning, I., Marris, L., Lever, G., Castaneda, A. G., Beattie, C., Rabinowitz, N. C., Morcos, A. S., Ruderman, A., Sonnerat, N., Green, T., Deason, L., Leibo, J. Z., Silver, D., Hassabis, D., Kavukcuoglu, K. and Graepel, T. (2018), 'Human-level performance in first-person multiplayer games with population-based deep reinforcement learning'.

Jorge, P. E. and Marques, P. A. M. (2012), 'Decision-making in pigeon flocks: a democratic view of leadership', *Journal of Experimental Biology* **215**(14), 2414–2417.
URL: *http://jeb.biologists.org/content/215/14/2414*

Kawabayashi, H. and Chen, Y. (2008), Interactive system of artificial fish school based on the extended boid model, *in* '2008 International Conference on Intelligent Information Hiding and Multimedia Signal Processing', pp. 721–724.

Reynolds, C. W. (1987), 'Flocks, herds and schools: A distributed behavioral model', *SIGGRAPH Comput. Graph.* **21**(4), 25–34.
URL: *http://doi.acm.org/10.1145/37402.37406*

Shiffman, D. (n.d.), 'Flocking'.
URL: *https://processing.org/examples/flocking.html*

Watts, I., Nagy, M., de Perera, T. B. and Biro, D. (2016), 'Misinformed leaders lose influence over pigeon flocks', *Biology Letters* **12**(9), 20160544.
URL: *https://royalsocietypublishing.org/doi/abs/10.1098/rsbl.2016.0544*