Exploration of the Boids algorithm for use in virus modelling

Research question: To what extent does the separation between boids in the Boids algorithm affect the rate of spread of a virus?

Subject: Computer Science

Word count: 3348

Table of Contents

Introduction

This essay will explore the Boids algorithm in the domain of virus modelling. Although originally created to generate flocks of birds using computer visual effects, the algorithm will now be used to model the spread of a virus throughout a population of boids that will represent humans.

First, the background of the algorithm will be explored, including its history and how it works. Furthermore, some basics of the science of the spread of disease will be covered, which will allow for the Boids algorithm to be justified as a model for the spread of disease. Most importantly, a variation of the Boids algorithm will be edited and used to perform primary research that will help answer the research question.

This research will come in the form of changing the intensity of one of the three essential rules of the algorithm, the separation between boids, whilst the rate of spread of the virus within the boid population will be measured. The separation was chosen as the variable to be changed because it can simulate the real-life effects of social distancing, and support its benefits. The results of this research will indicate a relationship between the two variables, and will be an example of how computer science can be used in other areas of research, such as that of virus modelling.

Background Information

History of the Boids model

The Boids algorithm was created by Craig Reynolds in 1986. Reynolds was employed at Symbolics Inc. and was working on computer visual effects. If a flock of birds was required in a shot in a film, they would usually be animated. Animating a massive number of birds was a laborious task, so Reynolds, who was always interested in the flocking motions of animals, developed an algorithm to facilitate this task by only requiring a model for a single bird. The name Boids comes from the play on words ‘bird-oid’. Reynolds’ research was published at SIGGRAPH in 1987 in a paper titled *Flocks, Herds, and Schools: A Distributed Behavioral Model.* (Failes, 2022)

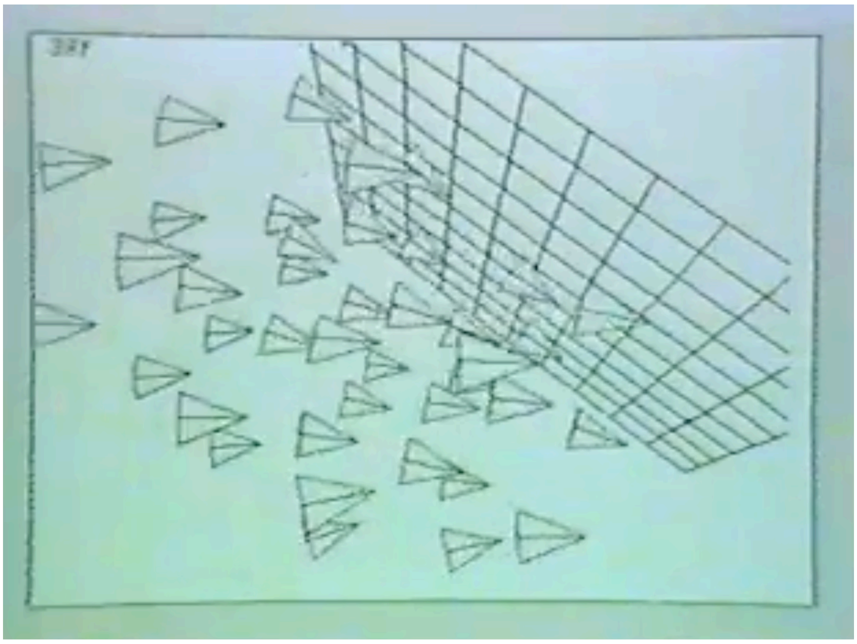


Figure 1. Original Boids simulation created by Craig Reynolds in 1986, which is 3-dimensional and features obstacles that the Boids avoid and manoeuvre around to then regroup. (ETJonasson, 2013)

The Boids algorithm was first used in an animated short film called *Stanley and Stella in: Breaking the Ice* (1987), and was most notably used later in the motion picture *Batman Returns* (1992) to animate swarms of bats, as they behave similarly to birds. It was also used in *The Lion King* (1994). (Failes, 2022)

Explanation of the algorithm

Boids models the overall behaviour of a flock of boids by determining the behaviour of each individual boid according to three rules that Reynolds came up with by closely observing real flocks. Those rules are the following:

Cohesion: each bird adjusts their direction of movement towards the average position of nearby birds, so that the boids stay in a group. If cohesion is the only rule applied, then all of the boids will merge into the same position. The cohesion of a boid (b_i) is calculated with two equations*.* In *equation 1,* the centre $(\vec{Fc}_l)$ of the flock (f) that b_i is in is calculated by summing the position $(\vec{p}_j)$ of each boid j (b_j) and dividing that sum by the total number of boids (N) in f. In *equation 2*, the cohesion vector $(\vec{Coh}_i)$ of b_i is calculated as the difference between $\vec{Fc}_l$ and the position of b_i. (Sanfilippo, 2014)

$$\vec{Fc}_l = \sum \forall b_j \in f \frac{\vec{p}_j}{N} \tag{1}$$

$$\vec{Coh}_i = \vec{Fc}_l - \vec{p}_i \tag{2}$$

Alignment: each boid adjusts its speed and direction of movement so that it equals the average alignment and speed of nearby boids, so that they move in unison. Without the alignment rule, the boids would bounce off of each other and not mimic the smooth flocking of real birds. The alignment is again calculated with two equations. In *equation 3,* the average velocity of nearby boids in the flock $(\vec{Fv}_1)$ is the sum of the velocity of the nearby boids $(\vec{v}_j)$ divided by N. In *equation 4,* the alignment vector $(\vec{Ali}_i)$ of b_i is the difference between $\vec{Fv}_1$ and the velocity of b_i. (Sanfilippo, 2014)

$$\vec{Fv}_l = \sum \forall b_j \in f \frac{\vec{v}_j}{N} \tag{3}$$

$$\vec{Ali}_i = \vec{Fv}_l - \vec{v}_i \tag{4}$$

Separation: each boid adjusts its position to maintain a certain amount of distance from nearby boids, to avoid overcrowding and collisions. If the separation rule is the only one that is applied, the boids would disperse and the flock would not form properly. This is rule uses *equation 5,* where the separation steer $(\vec{Sep}_i)$ of b_i is calculated as the negative sum of the difference between the position of b_i $(\vec{p}_i)$ and the position of the boids around it $(\vec{p}_j)$. (Sanfilippo, 2014)

$$\vec{Sep}_1 = -\sum_{\forall b_j \in f} (\vec{p}_i - \vec{p}_j) \tag{5}$$

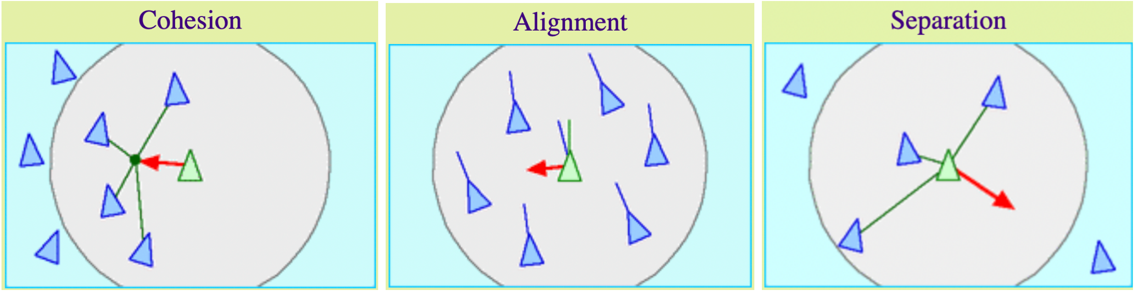


Figure 2: A visual representation of how the three rules of the algorithm affect the boids in a population. (Wong, 2008)

This simple set of rules allows for surprisingly complex behaviour. In a real flock of birds, it seems as though the ‘leader’ of the flock changes with every turn that the group makes. In this model there is similarly no fixed leader, as these three rules on their own quite accurately mimic a real flock. This is a key feature of a “swarm intelligence” system. Below is an example of the Boids algorithm as a whole.

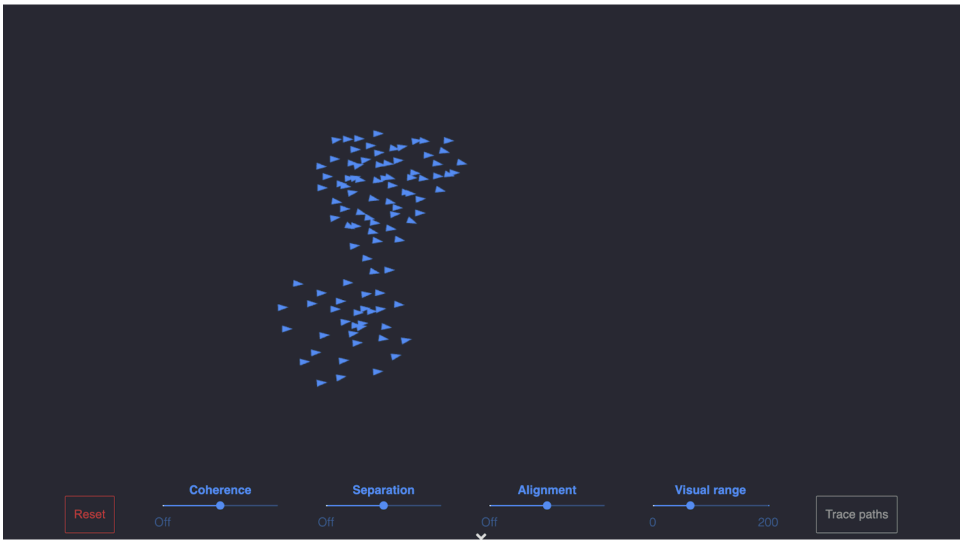


Figure 3. An interactive version of the Boids algorithm created by Ben Eater that allows the user to manipulate (by increasing or decreasing) the extent to which each of the rules/variables have an impact on the flock. (Eater, 2020)

Disease spread

As explained by the CDC, a way that many diseases spread, including measles and COVID-19 is via indirect airborne transmission with airborne particles. This often comes in the form of a sneeze, a cough, or even an exhale from an infected person whilst a healthy person is in the same space that these particles enter. This can very easily happen in a crowd of people. A healthy person will then become infected, and also be a source for the infectious agent and can pass it on to others. Some people are more susceptible than others, with factors such as age, genetics, and vaccination. Some ways to prevent infection are wearing masks, washing hands, and socially distancing. However, these strategies can only be executed if an infected person is aware that they are infected. (CDC, 2020)

Experiment Methodology

The Boids algorithm as a model for disease spread

a. Limitations of other models

A journal article on human biology called *Population Structure and the Spread of Disease* by Lisa Sattenspiel investigated models for the spread of disease. It begins by stating that many mathematical models make the incorrect assumption that all individuals in a host population mix randomly, which is the case for the model depicted below from the resource Shodor. (Sattenspiel, 1987)

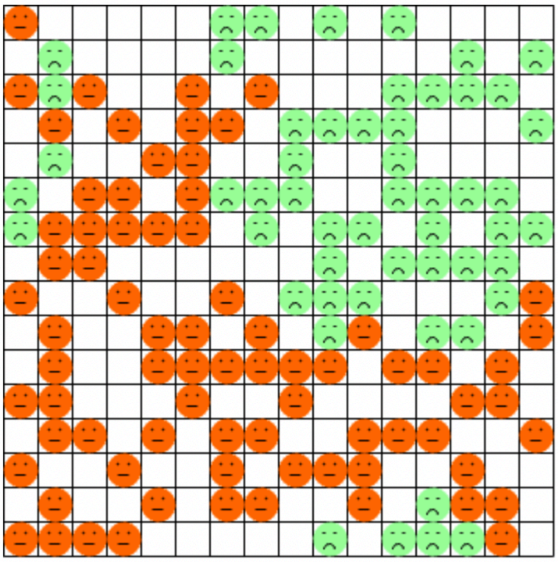


Figure 4. An interactive model from Shodor that has each circle (representing a person) changing squares in a random manner, with no clear direction. (Shodor, 2009)

The problem with this assumption is that individuals are usually distributed in space, which means that they will not mix randomly, but will rather have a pattern to their movement. A more accurate model will lead to the formation of multiple subgroups within the host population, where mixing might be more random within that group. This makes a big difference in that not all individuals will come into contact with each other. (Sattenspiel, 1987)

b. Suitability of the Boids algorithm

Boids was originally created to simulate a flock of birds, or a school of fish, but it can also represent a crowd of people. A central location for humans to walk past each other and come into contact, such as a train station or town centre, might look very similar to the Boids algorithm with many people walking in different directions, with groups sometimes moving side by side in the same direction.

Figure 5. Grand Central Station in New York City, depicting many people moving around in different directions. (Bounce, 2021)

This is a strength of the Boids algorithm. Just like humans, the boids will naturally form small groups, and within those the boids will move in a similar direction, until they run into another group. As a result of this, Boids is a more realistic model for the movement of a host population than most that can be found online, as it doesn't have completely random mixing thanks to its formation of subgroups. Boids will therefore be used to model the spread of disease through a group of people in this investigation.

c. Adaptation of the algorithm

The website Github features a community of programmers that post their open-source coding projects. On the website can be found a page created by user Ivan Hornung, called Pandemic Simulator. This is a variation of the Boids algorithm that features a disease within the population of boids.

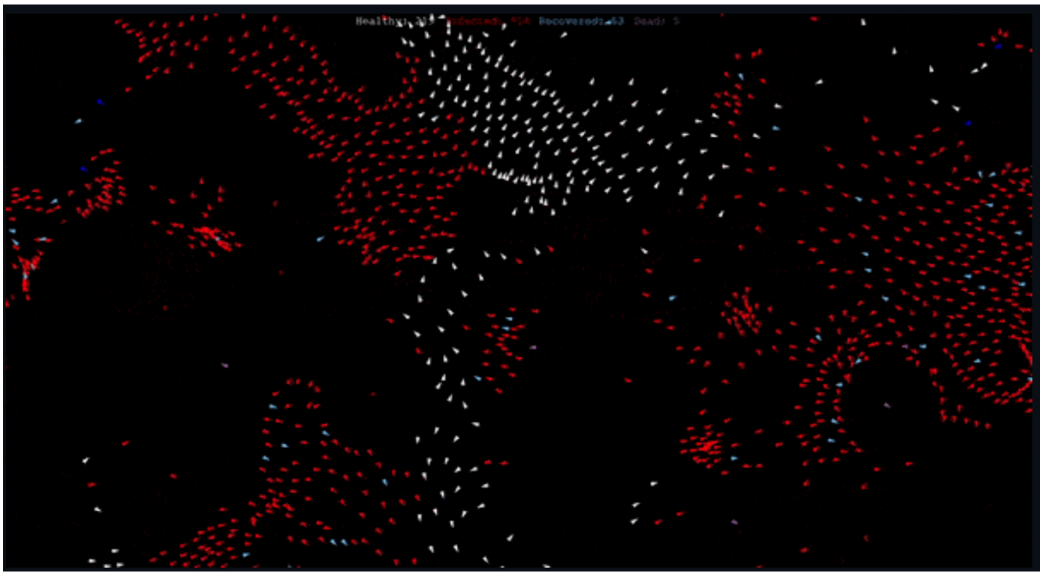


Figure 6. A screenshot of Ivan Hornung's variation of Boids. (Hornung, 2020)

This variation of the algorithm was created in 2020 during the COVID-19 pandemic. It features a number of 'types' of boids, that all have different roles in the host population. They are the following: healthy, infected, recovered, dead, diagnosed, paramedic, and paranoid. Each type has different behaviour and attributes within the population, with paramedics being able to cure infected boids, and recovered boids no longer being susceptible to the virus from infected boids, and more. *(Hornung, 2020)* For the purposes of the research question, this variation is exceedingly complex, as there would be too many uncontrolled variables at play that could influence the rate of spread of the virus throughout the population. Therefore, some changes were made to simplify the program, whilst retaining the vital feature of the virus within the population.

d. Changes made

The main change that was made was to remove all instances of any boids that weren't healthy or infected, which includes the recovered, dead, diagnosed, paramedic and paranoid. This was done to reduce the number of uncontrolled variables that might be introduced randomly into each trial. Not only that, but it allows me to measure the time that it takes for all boids in the population to become infected. This provides me with a measurable value for the rate of spread of the virus. The tradeoff of this is that the model becomes less realistic with fewer roles in the population, particularly by removing the recovered status. However, the opportunity cost is worth it in order to have data that can be analysed more precisely.

Some important information to note about this change is that Sattenspiel's research also stated that a fault in many models is ignoring the important element that is the recovered individual within a host population. The problem with not including it is that each infected individual remains infected for the entire simulation, and hence continuously contributes towards infecting the healthy individuals. In a real scenario, this would not happen, as people would build immunity to a virus, and therefore transmission would slow down. However, for the purposes of this research question, including the recovered boid status would be an impactful uncontrolled variable, so in this case the realism must be sacrificed for the effectiveness of this experiment.

Additionally, the code was edited to prevent the random introduction of new boids into the host population, which was originally a feature. This made it so the starting number of boids would remain the same (a predetermined number) for the entirety of the trial, and wouldn’t change from trial to trial.

Moreover, a new feature was implemented with the use of the java.time API. This is used to measure the amount of time it takes for something to occur, from the starting point, a certain event, to the ending point, another event. These were assigned with the Instant.now() function, and the time between the two events was calculated by the Duration.between() function. In the case of this experiment, the starting event is the moment the program is run, and the ending event is the moment that all of the boids in the host population have become infected. The duration of time between these events is output to the terminal in milliseconds. This API was implemented by following the official java.time documentation.

Other trivial changes involved changing the window size of the application, and removing all sounds and non-essential features.

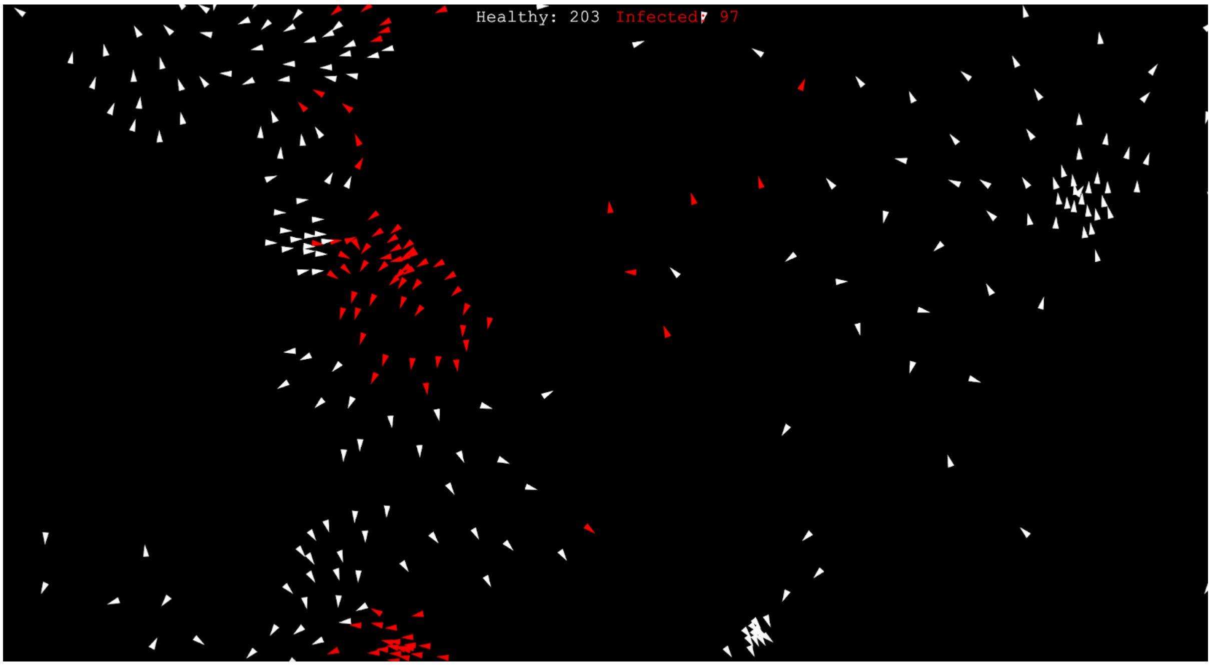
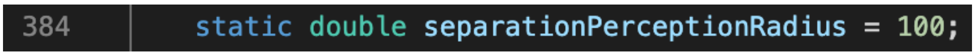


Figure 7. Ivan Hornung's Boids after the changes made were applied. (Hornung, 2020)

Method of collecting data using adapted algorithm

Variables

Independent variable: the separation between each boid, which is changed by editing the value of the variable named separationPerceptionRadius on line 384 of the Boid.java file. The settings are the following: 100, 150, 200, 250, 300. A larger value equates to more separation. The reason for these settings being chosen in that range is that a value less than 100 causes the boids to behave too strangely, as they form very tight-knit groups that don't accurately reflect the movement of people within a public space. The fact that these groups are formed, and that there is a fair amount of space within each group, means that despite the separation being so low, the rate of spread of the virus is actually low. This happens because it takes a long time for the groups to meet, whereas within each group, the virus spreads very quickly. On the other hand, having values above 300 causes the boids to almost freeze in place and not move, since any movement will mean that they enter the 300+ perception radius of another boid, and hence must turn in a different direction. Again, this behaviour is not realistic for people.



```
    static double separationPerceptionRadius = 100;
```

Figure 8. The separation perception radius static variable (that can be changed manually) in the code of the adapted Boids algorithm.

Dependent variable: the amount of time it takes for all boids in the host population to become infected. This is measured in milliseconds by the java.time API that is incorporated into the program. There will be 6 trials for each setting, for a total of 30 replicates.

Control variables:

- The number of boids in the host population. This number is set at 300, with 299 healthy boids and 1 infected boid.
- The intensity of the alignment and cohesion variables. This is controlled by leaving the alignmentPerceptionRadius and the cohesionPerceptionRadius at their default values set by the program's creator, at 50 and 100 respectively.
- The inclusion of only two types of boids: healthy and infected. This is controlled by removing any instance of other types of boids from the program.

The method for this experiment is that the program will be run for a certain separation setting. Once all boids become infected, the duration of time that it took will be output to the terminal in milliseconds. This duration will be recorded. This will be done a total of 6 times for each setting.

Dataset 1

Table 1. The results of changing the separation perception radius (s / $no\ units$) and measuring the time taken for all boids in the host population to become infected (t / ms)

	t						
s	T_1	T_2	T_3	T_4	T_5	T_6	Avg_t
100	23595	22975	23860	29973	30564	26402	26228
150	30970	26164	35077	24157	21346	23356	26845
200	20650	19799	19312	25674	23903	27140	22746
250	26303	32208	30427	31419	28323	29171	29642
300	75192	61908	54435	56280	44756	40264	55473

Table 2. The relationship between the separation perception radius (s / $no\ units$) and the rate of spread of the virus (r / $boids\ s^{-1}$)

s	r
100	11.4
150	11.2
200	13.2
250	10.1
300	5.4

Sample calculation for rate of spread of the disease

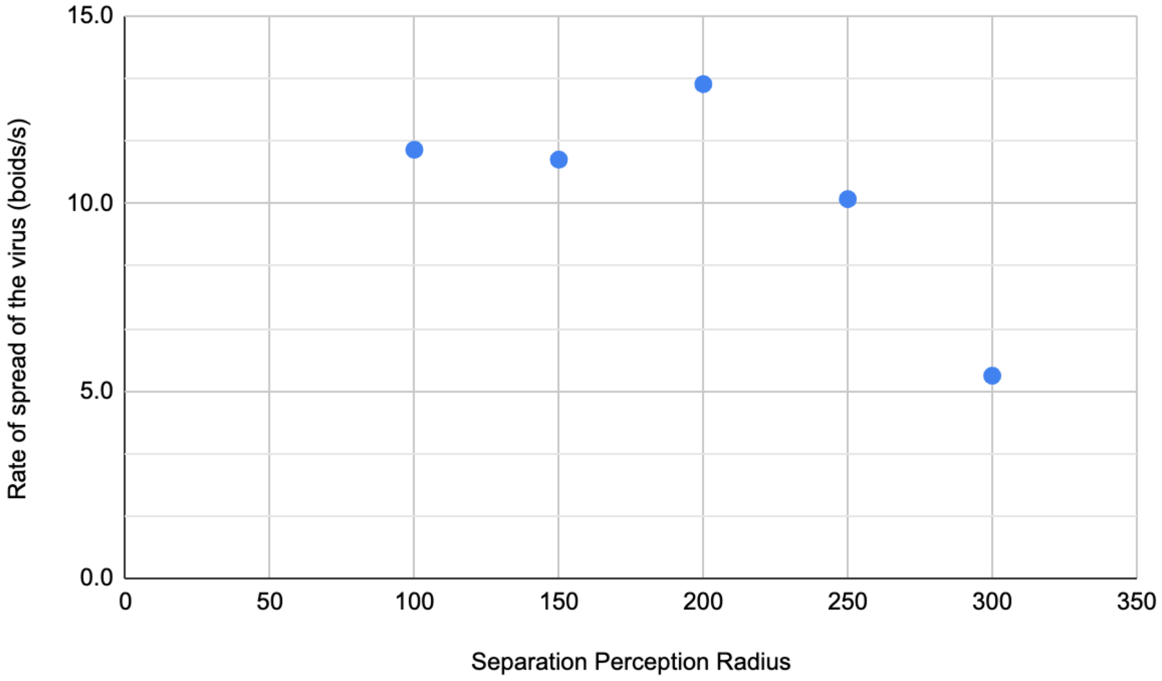
$r \ = \ 300 \,/\, Avg_t \times 1000$

Example for s = 100

$r = 300 \,/\, 26228 \times 1000$

$r = 11.4 \ boids\, s^{-1}$

Figure 9. How separation between boids affects the rate of spread of a virus



Dataset 1 analysis

Figure 9 displays a medium negative correlation between the separation perception radius of the boids in the population and the rate of spread of the virus within that population. This indicates that as the intensity of the separation rule increases, the rate of the spread of the virus decreases.

The separation perception radius values of 100, 150 and 250 follow this trend well, with a difference of 0.2 boids per second between radii 100 and 150, and a difference of 1.1 boids per second between radii 150 and 250, as can be seen in *Table 2*.

However, the radius of 200 was an outlier, as the rate of spread (13.2 boids per second) was quicker than all other radii, even the radius of 100. This happened because the boids behaved differently than they did with the other radii. This level of separation caused them to often switch between tight-knit groups and really dispersed crowds, which seems to have increased the spreading rate as more boids interacted with each other overall than smaller radii settings. It seems as though a radius of 200 was the tipping point between the more tight group-like behaviour of smaller radii settings and the more individually spread out behaviour of larger radii.

Furthermore, there is a big drop of 4.7 boids per second between radii 250 and 300. With radius 300, the separation was too large for the boids to move around without breaching the separation limit with other boids, so they almost stayed frozen in motion. Although this resulted in a very slow rate of spread, it doesn't accurately model the behaviour of real people moving around in a public setting.

Despite those outliers, there still seems to be a trend with the data. Consequently, a second set of data was collected with an additional independent variable: number of starting infected boids. In real life, there might be more than a single infected person within a crowd. This new data might reinforce or counter the trend in the first dataset. The settings for the number of starting infected boids are 2, 3. The number of total boids in the population will remain controlled at 300. The manipulation of the number of starting infected boids was another change that was made to the original code, where the number of additional infected boids (on top of 1) would be generated when the program is run.

Dataset 2

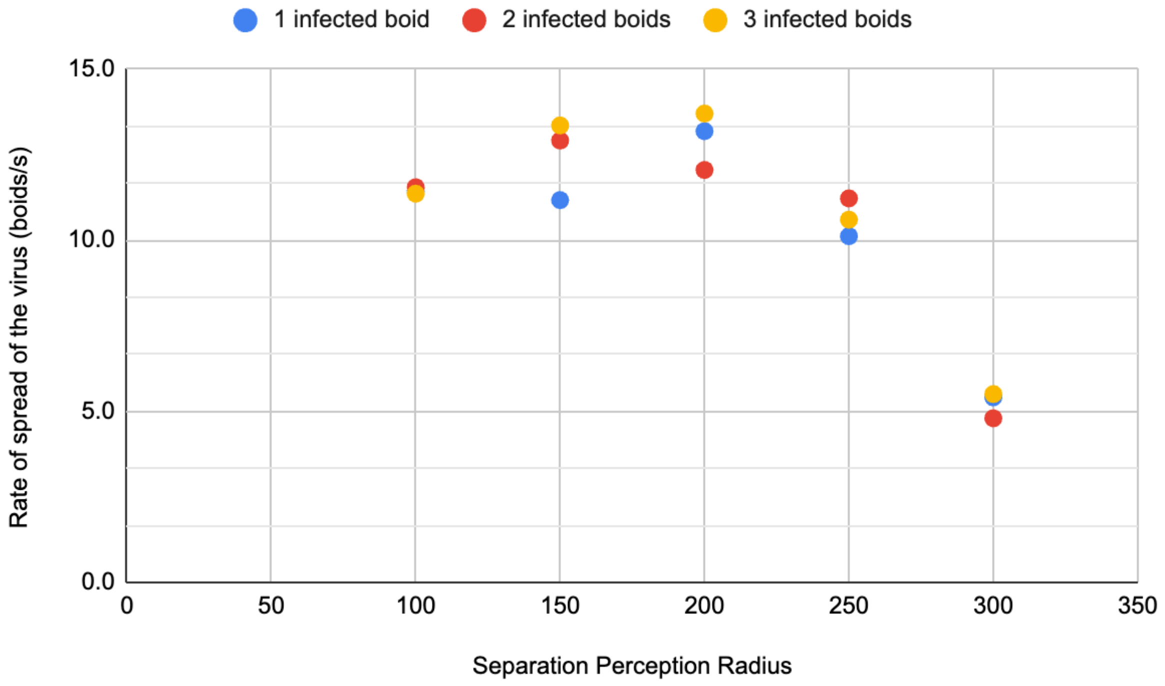
Table 3. The results of changing number of infected boids (n), and then changing the separation perception radius (s / $no\ units$) and measuring the time taken for all boids in the host population to become infected (t / ms)

		t						
n	s	T_1	T_2	T_3	T_4	T_5	T_6	Avg_t
2	100	23013	29842	22939	26260	28481	25293	25971
	150	24303	23263	27566	20724	21412	22091	23227
	200	20443	30477	35090	19400	17794	26110	24886
	250	22592	28175	29328	22198	30835	27250	26730
	300	61764	64821	70780	53819	60623	63429	62539
3	100	25089	23280	26085	37227	24021	22715	26403
	150	21123	24946	22989	23552	25707	16451	22461
	200	22531	17830	22396	22130	22962	23499	21891
	250	27082	28227	26740	27170	33872	26676	28295
	300	55964	67817	40544	63382	45527	53812	54508

Table 4. The relationship between the separation perception radius (s / $no\ units$) and the rate of spread of the virus (r / $boids\ s^{-1}$) for starting number of infected boids 1, 2, and 3

s	r_1	r_2	r_3
100	11.4	11.6	11.4
150	11.2	12.9	13.4
200	13.2	12.1	13.7
250	10.1	11.2	10.6
300	5.4	4.8	5.5

Figure 10. How separation between boids affects the rate of spread of a virus for different starting numbers of infected boids



Dataset 2 analysis

In *Figure 10*, it can be seen that by including 2 and then 3 starting infected boids, the trend is loosely followed. By looking at all three settings of infected boids, there is more of a weak negative correlation between the separation perception radius and the rate of spread of a virus.

The reason for why it has changed from medium to weak is that for 2 and 3 starting infected boids, there is an increase in rate of spread between 100 and 150. For 2 starting infected boids, it decreases between 150 and 200 radii but for 3 starting infected boids it increases further. It therefore indicates that increasing separation will initially also increase the rate of spread, but as the separation is increased further the rate of spread will decrease. This creates a sort of negative parabolic shape, in which a radius of 200 is the peak rate of spread/turning point of the function.

If the results from individual trials are examined in *Table 3*, it can be seen that the time taken for all of the boids to become infected varies greatly from trial to trial. For example, there is almost an 18 second difference between the shortest and longest times taken for the radius of 200 when there are 2 starting infected boids, which from about 17 seconds to 35 seconds is a 100% increase. A reason for this occurrence is that the effects of including more than one starting infected boid depended on the starting position of those boids. If infected boids generated near to each other within the host population, their impact wouldn’t differ much from only having 1 infected boid in their place, since the boids in that area would be infected at a similar rate. However, if the infected boids generated far away from each other, they would infect the whole population quicker because they would be infecting different parts of the population at the same time. This is an impactful uncontrollable variable in the methodology. Nevertheless, the overall trend is still faintly visible.

Conclusion

In conclusion, the results from the research indicate that the separation between boids in the Boids algorithm affects the rate of spread of a virus to a lesser extent, with inconsistent results that only lead to a weak negative correlation. This suggests that perhaps the algorithm is not the most suitable model for the spread of a virus, and/or that there are more factors involved in the outcome of the rate of spread. Both might be the case, which could be a consequence of removing key players from the simulation such as the recovered boids.

The outlier of the separation perception radius of 200 that was prevalent in both datasets, as well as the unnatural behaviour caused by the radius of 300, prompts the question of whether there is an ‘optimal’ separation perception radius between those values. This value could perhaps result in a relatively low rate of spread of disease, whilst still enabling the boids to move around naturally, as real people need to. This could reflect the real life distance of 2 metres that was important during COVID-19, since that was what was recommended by hospitals as the minimum distance that people should keep from each other to minimise the spread of the virus. Despite its inconvenience, it still allowed for people to move around in public places.

Overall, the Boids model still indicates that social distancing is a relatively effective way of reducing the spread of a virus within a population. Should the simulation be extended, by including key players as well as including other behaviours that are important during a virus outbreak such as wearing masks, and even quarantining (with faster response rate to infection leading to a slower rate of infection afterwards), the results might be more clear and have a stronger negative correlation. This investigation suggests that computer science has a place within virus research, and could be effective at modelling viruses among other phenomena provided sufficient features are involved.

Bibliography

Alaliyat, S., Yndestad, H., & Sanfilippo, F. (2014). Optimisation Of Boids Swarm Model Based On Genetic Algorithm And Particle Swarm Optimisation Algorithm... *Proceedings of the 28th European Conference on Modelling and Simulation (ECMS)*, 2–3. https://www.researchgate.net/publication/268077894_Optimisation_Of_Boids_Swarm_Model_Based_On_Genetic_Algorithm_And_Particle_Swarm_Optimisation_Algorithm_Comparative_Study

Bounce. (2021, April 5). *Grand Central Station Visitor Guide 2024: Everything you need to know*. Bounce. https://usebounce.com/guides/new-york/grand-central/visitor

Centers for Disease Control and Prevention. (2022, March 1). *How Does Disease Spread*. CDC. https://www.cdc.gov/scienceambassador/videos/how-does-disease-spread-quick-learn-transcript.pdf

Eater, B. (2020, April 27). *Boids algorithm*. Eater.net. https://eater.net/boids

ETJonasson. (2013). Craig Reynolds - Original 1986 Boids simulation [YouTube Video]. In *YouTube*. https://www.youtube.com/watch?v=86iQiV3-3IA

Failes, I. (2022, March 10). Befores & Afters – Issue #4. *Befores & Afters*, *4*, 22–30.

Hornung, I. (2020, April 29). *IvanHornung/Pandemic-Simulator*. GitHub. https://github.com/IvanHornung/Pandemic-Simulator

java.time (Java Platform SE 8). (2024, June 5). Oracle.com. https://docs.oracle.com/javase/8/docs/api/java/time/package-summary.html

Sattenspiel, L. (1987). Population Structure and the Spread of Disease. *Human Biology*, *59*(3), 411–438. Jstor. https://www.jstor.org/stable/41464816?seq=1

Shodor. (2009, August 4). *Interactivate: Spread of Disease*. Shodor.org. http://www.shodor.org/interactivate/activities/SpreadofDisease/

Wong, T. (2024, September 15). *Boids*. Stanford.edu. https://cs.stanford.edu/people/eroberts/courses/soco/projects/2008-09/modeling-natural-systems/boids.html