ENGG 3130 Case Study: Autonomous Drone Simulations using Boids

Liam Holt
University of Guelph
Guelph, Canada
holtl@uoguelph.ca

Kobe Barrette
University of Guelph
Guelph, Canada
kbarre05@uoguelph.ca

Klaire McCarthy
University of Guelph
Guelph, Canada
klaire@uoguelph.ca

Thomas Zupan
University of Guelph
Guelph, Canada
tzupan@uoguelph.ca

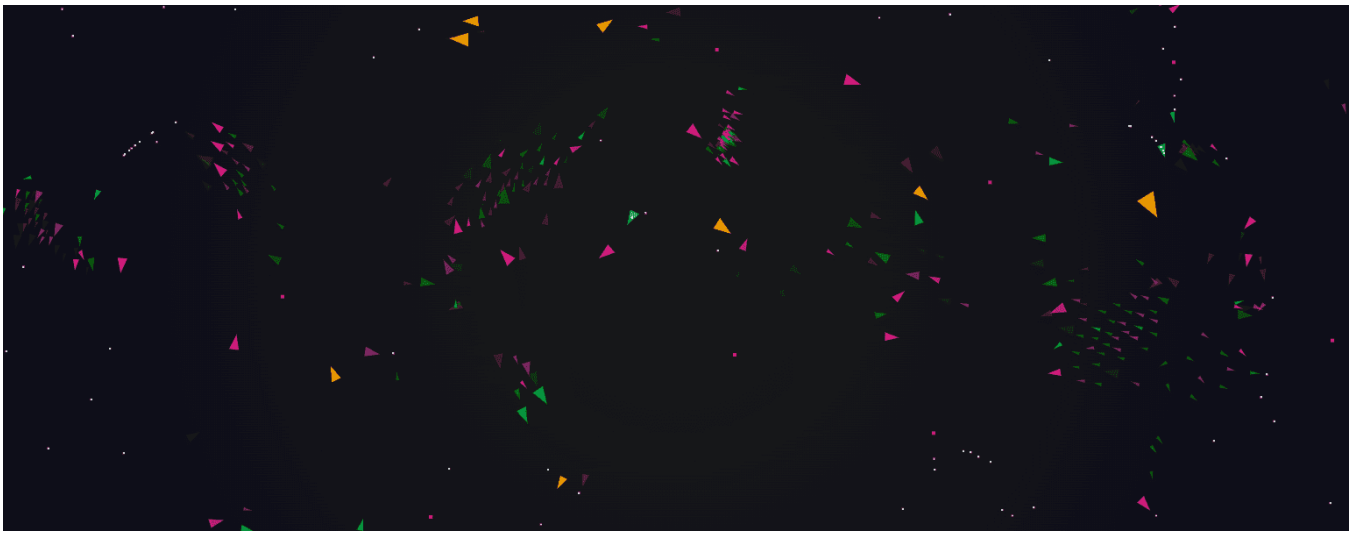

Figure 1: Boids: A glimpse into artificial life and collective animal behavior [1].

Abstract

Drone behavior in search and rescue applications has been explored by comparing the behavior of Boids in a single swarm versus multiple swarms. The success of each swarm has been measured by the speed with which it reaches its target. For both swarm types, the response time was measured for a fixed target, a random target, multiple targets, and an object to avoid. The Boid algorithm was based on Allen Downey's algorithm in Think Complexity[4].

CCS Concepts

• **Computing methodologies** → Multi-agent systems; Agent / discrete models; Artificial Intelligence; • **Computer systems organization** → Robotic autonomy.

Keywords

Boids, Flocking, Swarm Intelligence, Multi-Agent Systems, Unmanned Aerial Vehicles (UAVs) or Drones, Search and Rescue.

ENGG 3130 Case Study, University of Guelph
2025.

1 Introduction

This case study investigates the effectiveness of The Boid Algorithm to model the collision avoidance and target location of swarms of autonomous unmanned aircraft in a search and rescue scenario. The investigation begins with the Boid class implementation from Allen Downey's Think Complexity[4]. This study introduces a performance metric that measures how quickly a target is located/reached by the drones. This metric was used to quantify the effects of varying system parameters and introducing new objects and obstacles to a simulated rescue operation.

A baseline simulation was established with a fixed target location and one swarm of search drones. Beyond the baseline, the package location was then randomly selected, and more than one package and avoidable objects were introduced to the search ground. The effect on rescue time was evaluated. The second part of the investigation measured the effectiveness of a secondary swarm with the same variations that were introduced in the first section.

The goal of this investigation is to discover relationships between the rescue time of the swarm and the various changes to the situation. This can identify where optimizations need to be made

when implementing a similar search and rescue system. A continuation of this study would implement a drone swarm rescue system and evaluate if the results reflect what the Boid model predicts. An autonomous collision avoidance system can also be implemented into drone navigation systems for many additional applications.

2 Problem Description

Autonomous swarms of unmanned aircrafts can be used in a search and rescue operation in many contexts including natural disasters, warzones, and other incidents [9]. Minimizing the time it takes to locate a victim is crucial to improving rescue efficiency and survival rates in these incidents. To be able to do so, it is important to identify what environmental conditions and obstacles will impede rescue time the most.

2.1 Motivation and Objectives

This study will explore a method in simulating a search and rescue operation conducted by a swarm of unmanned aircraft. The boid algorithm can effectively simulate flock behaviour and is commonly used to simulate animal behaviour such as a flock of birds or a school of fish. This study is an extension of these applications.

2.2 Boid Model and Its Application to Swarm Search and Rescue

The boid model, originally developed by Craig Reynolds, is an artificial life simulation [11]. The algorithm implements agents that have four main behaviours. The first behaviour, flock centering has the agent move towards the center of the flock. This is done by calculating a vector that points towards the centroid of the flock [4]. The second behaviour, avoid, aims to determine what other agents are within a range and calculates a vector that points away from their centroid to not collide with other agents [4]. Align, the third behaviour calculates the headings of nearby agents, creates an average vector of the headings and uses that vector as the agents new heading [4]. The final behviour, love, is used to direct agents towards a carrot (target location) by calculating a vector that points towards the desired location [4]. This algorithm was selected to model this problem as it mimics extremely similar behaviour that is needed in a swarm search and recue environment. The drones must avoid collisions with other drones, remain in the same heading, and direct themselves towards a target location.

3 Related Work

3.1 Object Avoidance in Boid Algorithms

As discussed in "Extending boids for safety-critical search and rescue" by Cole Hengstebeck et al. [5], the fault in the boids algorithm is that it does not address object avoidance. Boids in a swarm fly into and over each other without consequence, when in reality a collision would result in boid injury or death. It is an additional challenge to design a system that physically performs well and also has strong computational ability; high processing power often comes at the cost of a lightweight system. In the model described by Hengstebeck, object avoidance is addressed by adding "ghost boids", which are regular boids stationed relative to various objects, and are meant to modify the behaviour of the swarm boids. The ghost boids do not affect the three factors of boid behaviour: separation, cohesion, and alignment. The "aids-to-navigation" ghost boids were stationed around the boundary, so that if a swarm boid encountered it, they would turn away. The aids-to-navigation boids also redirected the swarm boid's head away from the boundary, thereby avoiding danger. The "compass" ghost boids are placed directly atop the swarm boids, modifying the alignment rule of its agent by pointing its heading toward the target. The authors have provided an alternate method to manage the lack of object avoidance in boids.

3.2 Boid Modelling in Aquatic Systems

While boids originated by modelling bird flocking behaviour, it can be used to model behaviour of other animals who exist in large groups, such as schools of fish. There is considerable research into biomimicry of fish. From a programming angle, the boid algorithm has been used to study school behaviour in different scenarios such as net fishing [10]. Robotic systems research explores the physical system, including what materials from which caudal fins should be created [13]. Research is well on its way to fully mechanized fish, whose live counterparts follow boid behaviour.

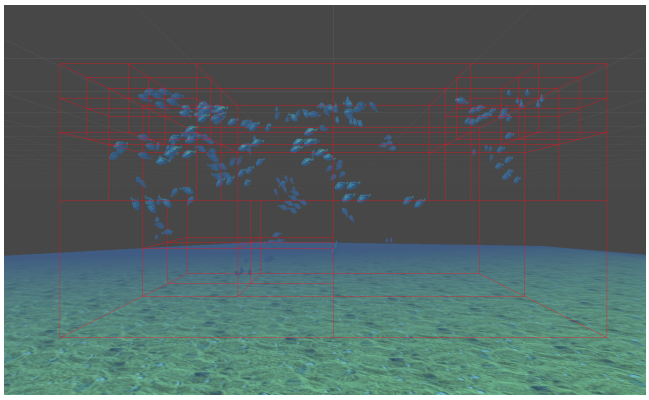

Figure 2: Example of aquatic Boids simulation for a school of fish [2].

3.3 Boid-Inspired Behavior in Social Media Networks

The behaviour of likes on social media posts in reference to boid behaviour [6] has also been investigated. The authors found that for a social media post, once a threshold of "total number of users" has been reached, the post becomes easier to find, i.e., it has a higher chance of appearing in users' feeds. This is because new tags are created on the post once that threshold is reached. For a group of users with similar interests, this threshold is reached faster. The authors link this activity to boids, where a large group of users with similar interests adopts the behaviour of a large flock. In a large flock, there exists an element of randomness to each individual's behaviour, despite the overall flock behaviour being uniform. This behaviour is consistent with a large group of users—they may have a common interest in a certain genre of posts, but each individual

also sees posts tailored to their own interests, which would be considered "random" from the group.

3.4 Application of Boid Algorithms in Autonomous Drone Systems

There has also been movement on the topic of boids and drones. There are multiple papers discussing the feasibility [7] and potential designs [8] of drones using the boid model for unmanned aerial vehicles, which are acknowledged to be beneficial for search and rescue or in military applications [3]. Autonomous drones can also be used for improved environmental monitoring, such as for wildfires [12]. Since a pack of drones would create a wireless sensor network, a new routing protocol for a pack of autonomous drones has been proposed [3].

4 Methodology

4.1 Initializing the simulation space

The investigation began with the boid and world classes and associated functions from the ENGG3130 and Think Complexity notebooks. The first change was the implementation of the simulation space. A wireframe cube was used to represent an outdoor search and rescue scene. Trees, grass, and clouds were added to enhance the environment and improve the aesthetic quality of the simulations. The simulation space can be seen in Figure 3.

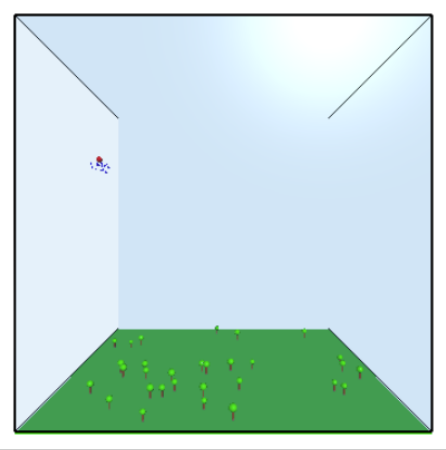

Figure 3: Simulation space setup.

The drones are represented as cones and the package, or carrot, is a sphere. These can be seen clearly in Figure 4.

4.2 Performance Metrics

The first simulation included one package placed in the center of the simulation space and the boids began in the bottom left corner. To quantify the results, the time it takes for a boid to reach the package was measured. On average, in this scenario, the boids took 9.5 seconds to reach the package.

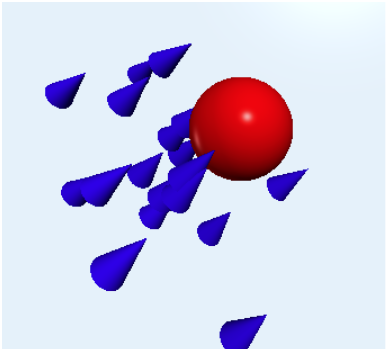

Figure 4: Close up of Boid/Drone and Package/Carrot.

4.3 Adding a Second Carrot

The next simulation introduced an additional condition to make the situation more realistic. The carrot was placed in a random location. The time it takes to reach the carrot is proportional to how far the carrot is from the starting position of the boids. This was often around 10 seconds. The third simulation added a second carrot to simulate a situation where there are multiple targets that the drones need to reach. This was done by inheriting from the original boid class and making necessary modifications to have the boids seek the second carrot. The world class was also modified to include two carrots as well as ensure that the second carrot does not spawn too close to the first. Figure 5 shows a simulation space that contains two carrots. Due to the minimum distance between the

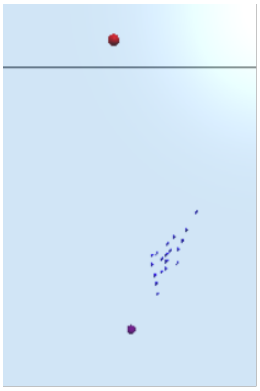

Figure 5: Simulation space with two carrots.

two carrots, the second carrot was typically reached in an additional four seconds if the distance was small, and a maximum of around ten seconds if the distance was larger.

4.4 Introducing Obstacles

The next step in the investigation was to introduce obstacles into the environment. To do so, the boid class was again modified to be able to avoid the obstacles. The world class was also modified to handle the two carrots and the new obstacles. Figure 6 shows a simulation space with three obstacles. Running the simulation with three randomly placed obstacles and the two carrots led to

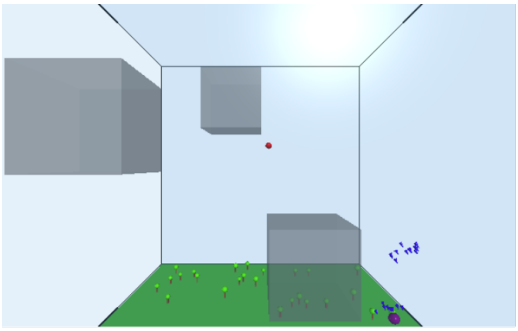

Figure 6: Simulation space with two carrots and obstacles.

similar results as the simulations without the obstacles. This is due to the large simulation space that allows the boids to take quite direct roots that avoid obstacles. This is representative of the real system as in a search and rescue environment that is outdoors there would be typically ample room to avoid obstacles. However, when the obstacles were in certain locations, they did increase the time it took for the boids to reach the carrot.

4.5 Introducing a Second Swarm

The next step was to introduce a second swarm of boids to the simulation space. This was done by creating a boid class based on the boid class that avoids obstacles. However, this class will spawn the boids in the opposite corner and have them be a different colour. Additionally, the world class was modified to handle the second swarm and introduce random obstacle shapes. Figure 7 shows a simulation space with three obstacles, two carrots and two swarms of boids.

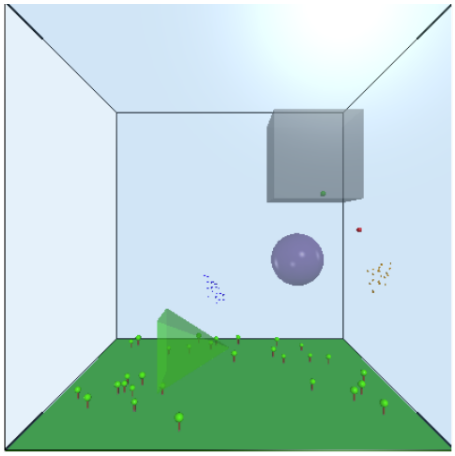

Figure 7: Simulation space with two carrots, obstacles and a second swarm.

4.6 Quantifying the Results

To quantify the results, a set of 5 trials were run with the above scenario. The time to reach each carrot by each swarm was plotted and can be seen in Figures 8 and 9.

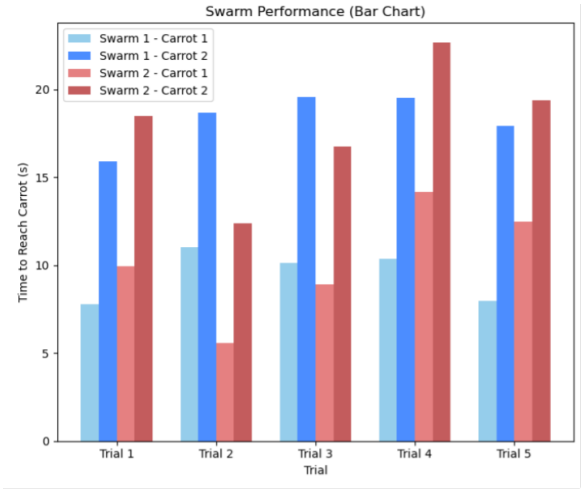

Figure 8: Bar chart with results from 5 trials.

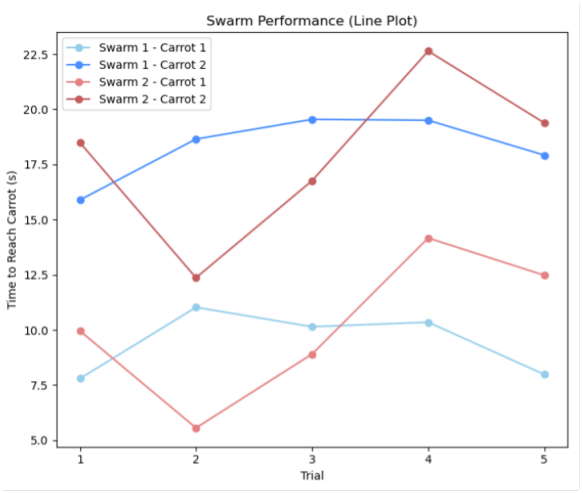

Figure 9: Line chart with results from 5 trials.

Table 1: Time for each swarm per trial.

Trial	Swarm 1 (Total Time)	Swarm 2 (Total Time)
1	16.0s	18.0s
2	18.0s	12.5s
3	19.5s	16.5s
4	19.2s	22.5s
5	18.0s	20.0s

The results show that it took consistently longer for the swarms to reach the second carrot (dark blue and red lines).

5 Discussion

5.1 Single Swarm Behavior with Randomly Placed and Fixed Packages

In a single swarm, a randomly placed object takes the same amount of time to find as a fixed object, which is beneficial to know, given that search and rescue situations feature targets at completely random locations. Since many search and rescue applications often involve victims stuck in one place, it is reasonable for the package to be fixed in space once it is placed. It was observed that two packages placed nearby each other were found in about 1.5 times the time it takes to find a single package. Two packages placed very far apart from each other acted like two instances of a single package; it took about double the time to find both packages, since they were far apart. Given obstacles, ample space allows the swarm to behave normally and there is little to no hinderance to finding an obstacle.

5.2 Effects of Multiple Packages and Obstacle Proximity on Search Efficiency

Further investigation into single swarm behaviour should explore the addition of more than two packages, and more obstacles closer together. Presently, there is a reciprocal relationship between the number of packages and the time it takes find them after the first package has been found; it took half the time to find the second package nearby. Presently, it cannot be determined whether this is a geometric or an arithmetic relationship due to lack of data. Therefore, exploring the behaviour of a third (or fourth, or fifth, etc.) package nearby would provide insight as to whether the second and third packages would each be found in one-third of the time it took to find the first package, or if it takes half the time of the first package for each additional package. Second, adding obstacles closer together may alter the behaviour of the swarm and should therefore be explored as well. The limited space may decrease the time to find a package because there are fewer places in which it could be located, or it could increase the time because the boids must avoid collisions with each other and the obstacles.

5.3 Multiswarm Behavior Compared to Single Swarm Behavior

In a multiswarm, the behaviour of a fixed package, randomly placed package, and multiple randomly placed packages are consistent with the behaviour of a single swarm. However, once obstacles were introduced, which is the most realistic model for search and rescue applications, it took significantly longer for a multiswarm to find both packages than for a single swarm to find them. It could be possible that there is a critical swarm number, below which one swarm is more effective than multiple swarms, and above which multiple swarms are more effective than a single swarm. However, if this number is large, it is possible that the multiswarm will behave as a very big single swarm. This behaviour should be investigated further.

5.4 Practical Recommendations for Search and Rescue Operations

For search and rescue applications, a single swarm should be used in one area, according to the discussion above. However, depending on the size of the physical area being searched and physical constraints such as speed and battery life of the drones in operation, it may also be beneficial for a single swarm of drones to be confined to one geographic region. By breaking a large region into segments, a single swarm of drones could search its segment effectively, and second single swarm of drones could search a second segment, and so on. However, this hypothesis of multiple single swarms versus multiple swarms in one region should be investigated as well.

References

[1] [n. d.]. Flocking Simulation. https://thecodingtrain.com/challenges/124-flocking-simulation. Accessed: 2025-04-19.

[2] Albjoh and Gupett. [n. d.]. Working Boid Simulation with Octrees — gupett.github.io. https://gupett.github.io/CG-Project/. [Accessed 19-04-2025].

[3] Nour El Houda Bahloul, Saadi Boudjit, Marwen Abdennebi, and Djallel Eddine Boubiche. 2018. A Flocking-Based on Demand Routing Protocol for Unmanned Aerial Vehicles. *J. Comput. Sci. Technol.* 33, 2 (2018), 263–276.

[4] Allen B. Downey. 2016. *Think Complexity*. Green Tea Press, 9 Washburn Ave. Needham, MA, 02492.

[5] Cole Hengstebeck, Peter Jamieson, and Bryan Van Scoy. 2024. Extending boids for safety-critical search and rescue. *Franklin Open* 8 (2024), 100160.

[6] Takashi Ikegami, Yoh ichi Mototake, Shintaro Kobori, Mizuki Oka, and Yasuhiro Hashimoto. 2017. Life as an emergent phenomenon: studies from a large-scale boid simulation and web data. *Philos. Trans. R. Soc. Lond.* 375, 2109 (2017), 20160351–20160351.

[7] Brett M. Martin, Ryan D. Winz, Luke J. McFadden, and Tor J. Langehaug. 2023. Distributed Boids Simulation: Performance Analysis and Implementation Challenges. In *2023 Congress in Computer Science, Computer Engineering, & Applied Computing (CSCE)*. IEEE, 780–785.

[8] Adam Pooley, Max Gao, Arushi Sharma, Sachi Barnaby, Yu Gu, and Jason Gross. 2023. Analysis of UAV Thermal Soaring via Hawk-Inspired Swarm Interaction. *Biomimetics* 8, 1 (2023), 124.

[9] Anam Tahir, Jari Böling, Mohammad-Hashem Haghbayan, Hannu T. Toivonen, and Juha Plosila. 2019. Swarms of Unmanned Aerial Vehicles — A Survey. *J. Ind. Inf. Integr.* 16 (2019), 100106.

[10] Yuki Takahashi and Kazuyoshi Komeyama. 2020. Simulation of the capture process in set net fishing using a fish-schooling behavior model. *Fisheries Sci.* 86, 6 (2020), 971–983.

[11] Timm[ie] Wong. 2008. Boids. https://cs.stanford.edu/people/eroberts/courses/soco/projects/2008-09/modeling-natural-systems/boids.html. Accessed: 2025-04-17.

[12] Qingli Zeng, Harir Razzazi, and Farid Nait-Abdesselam. 2024. Cooperative and Autonomous Flocking of Drones Using an Extended BOID Model. In *IEEE Global Communications Conference (Online)*. IEEE, 1311–1316.

[13] Chen Zheng, S. Shatara, and Tan Xiaobo. 2010. Modeling of Biomimetic Robotic Fish Propelled by An Ionic Polymer-Metal Composite Caudal Fin. *IEEE/ASME Trans. Mechatronics* 15, 3 (2010), 448–459.

AI Tools Used

- ChatGPT: Used to understand original Boid code and help expand simulations. Was also used to help provide feedback on the code and debugging purposes.
- Perplexity.ai: Used to help setup LaTeX environment and find appropriate commands.