TEB1113/TFB2023: Algorithm and Data Structure

September 2024

Group Name: Synchrotech

Guided Project

Prepared for: Dr. Nordin Zakaria

Name	Student ID	Programme
Siti Nurfatimah Az Zahra Binti Norhisham	20001348	Computer Science
Yasreen bt Mohamed Yusoff	22005648	Computer Engineering
Nurul Anisa binti Sufian	22005637	Computer Engineering
Nur Husna Husniyah binti Abdul Razi	22002729	Computer Engineering
Addly Aiman bin Mohamad Faizal	22004410	Computer Engineering

Table of Contents

1. Introduction

In the rapidly evolving field of unmanned aerial systems, drone swarms have emerged as a fascinating area of research and development. This project aims to delve into the intricacies of simulating drone swarm behavior, with a particular focus on comparing two distinct implementation approaches: a from-scratch method and a library-based solution. By exploring these contrasting techniques, we seek to uncover valuable insights into the efficiency, scalability, and overall performance of different simulation strategies. Our objectives include not only the successful replication of swarm intelligence algorithms but also a comprehensive analysis of the trade-offs between custom-built solutions and pre-existing frameworks in the context of Unity-based simulations.

To lay the groundwork for our project, we delved into extensive background research focusing on two key areas: swarm algorithms and Unity game engine. We explored several pivotal swarm algorithms, including Boids, Leader-Follower, and Particle Swarm Optimization (PSO), each offering unique approaches to coordinating collective behavior. These algorithms form the backbone of our simulation, providing the theoretical framework for drone interactions. Simultaneously, we investigated Unity as our chosen development platform, attracted by its robust 3D capabilities, physics engine, and extensive documentation. Unity's ecosystem, particularly its AI-focused libraries and packages, presented promising avenues for implementing complex swarm behaviours efficiently.

2. Background Research

2.1 Implementation in C# and GitHub

C# Implementation

The simulation of drone swarm behavior was implemented in Unity using C# scripts to control the drones' movement and interactions. Two approaches were utilized: a from-scratch implementation of the Boids algorithm and a library-based approach using Unity's ML-Agents package.

1. From-Scratch Approach (Boids Algorithm):

- In this approach, we manually coded the three main behaviors of a drone swarm: cohesion, separation, and alignment.
- Cohesion ensures drones move towards the average position of their neighbors.
- Separation prevents drones from crowding by making them steer away from nearby drones.
- Alignment makes drones match the direction and speed of their neighbors.
- The behavior was implemented using a single script attached to each drone object in Unity, updating each drone's velocity and position in every frame.

2. Library-Based Approach (Unity ML-Agents):

- For the library-based implementation, we used Unity's ML-Agents toolkit, which allows for reinforcement learning-based simulations.
- Drones were trained as agents to learn how to move towards target points while avoiding obstacles and coordinating with each other.
- The ML-Agents package handles learning behavior by training agents using machine learning algorithms, and the agents adapt their behavior over time.

Both approaches allowed us to simulate realistic drone swarming behaviors with varying levels of complexity and performance. The comparison between the manual implementation and the library-based implementation helps us to evaluate the trade-offs between development effort and simulation performance.

GitHub Repository Setup

To streamline collaboration and version control, the project was hosted on GitHub. The following steps were followed for setting up the repository:

1. Repository Creation:

- A new private GitHub repository was created, ensuring that all team members could contribute to the codebase. The repository was set up to track changes in the C# code, Unity project files, and configuration settings.
- The GitHub repository can be accessed via the following link: `[Insert GitHub Link]`.

2. Branching and Collaboration:

- A branching strategy was adopted to manage different stages of development. The main branch contains the stable version of the project, while feature branches were created for each task (e.g., Boids implementation, ML-Agent integration).
- Each member worked on separate branches and submitted pull requests for code review, ensuring smooth integration of new features into the main branch.

3. Code Documentation and Version Control:

- All scripts were documented using in-line comments to provide clarity on key functions and algorithms.
- GitHub Issues were used to track bugs, new features, and tasks, ensuring that the team could monitor project progress effectively.
- Regular commits were made to ensure proper version control, allowing us to roll back changes if necessary.

By using GitHub for collaboration, we ensured a structured and efficient workflow for the development and integration of both the from-scratch and library-based swarm algorithms.

2.2 Swarm Algorithms

Swarm algorithms are inspired by natural behaviors, such as bird flocking and fish schooling, where agents operate based on simple rules to achieve complex group behaviors. The following algorithms are frequently used in drone swarm simulations:

- Boids Algorithm: A rule-based system developed by Craig Reynolds, which governs agent behavior based on cohesion, separation, and alignment.
- Leader-Follower Algorithm: This approach designates one or more drones as leaders, with the remaining drones following their movement.
- Particle Swarm Optimization (PSO): A population-based optimization algorithm inspired by the social behavior of birds, where agents (drones) share information to optimize their position in a search space.

2.3 Unity

Unity is a powerful game engine widely used for simulating real-time 3D environments. It is chosen for this project due to its flexibility, real-time rendering capabilities, and compatibility with various simulation algorithms. Unity's ability to integrate custom scripts allows for the manual implementation of swarm behaviors, while its package manager provides access to robust libraries such as ML-Agents, enabling complex machine learning-driven behaviors in simulations.

3. Methodology

- Simulation Design & From Scratch Approach

From Scratch Approach involved manually coding and developing the core algorithm to control the behaviour of the drone swarm without relying on external libraries. It focuses on flocking using Craig Reynolds' Boids model, which involves separation, alignment, and cohesion. These rules ensure drones avoid collisions, move in same direction, and stay grouped. The Leader-Follower model is used to simulate more structured swarm behaviour. In this concept, a selected leader drone moves along a set route, while the rest of the swarm changes their movement based on the leader's position and trajectory. This method was coded to allow the swarm to take difficult routes while maintaining formation. Implementing these algorithms from scratch was optimizing the system to handle large numbers of drones while maintaining smooth performance. The computational complexity is reduced by limiting the number of neighbouring drones each one considers in the Boids and Leader-Follower models.

- Library-Based Approach

A library-Based Approach utilizes the library that is provided by the programs. The codes include Unity ML Agent toolkit to control the drone and navigate it to reach the target. Four libraries will be imported which are Unity ML Agents, sensors, actuators and Unity Engine. Unity ML Agents consists of ML functionalities like agents and environments. Unity ML Agent sensors is used to gather observations from the environments while Unity ML Agent actuator is for handling actions. Unity Engine is the core Unity functionalities. The drones will learn how to navigate around to reach the target. Along the way, observations abouts its position, the targets and the orientation will occasionally be reported back to the drone. The information received by the drones are used to propel them forward or backward and once they reach the target, it will be rewarded. Training loop will be added into the code so it can reset both the drone and target each time.

4. Empirical Runtime Performance Study

- **Metrics:**
    - Execution Time: We will measure how long each approach takes to simulate swarm behavior over a specified period.
    - Memory Usage: We will monitor memory consumption as the number of drones in the simulation increases.
    - Scalability: We plan to assess how both approaches scale with varying numbers of drones (e.g., 10, 100, 500) and how performance changes as complexity grows.
- **Procedure:**
    - For each approach (from-scratch and library-based), we will run simulations with different numbers of drones under identical environmental conditions and behaviors.
    - We will use Unity's profiling tools to collect runtime data, specifically tracking execution time and memory usage.
    - We will evaluate scalability by observing how each approach manages increasing drone counts without significant performance degradation.
- **Data Collection and Visualization:**
    - We will tabulate data for execution time and memory usage across different scenarios, comparing from-scratch and library-based approaches.
    - Graphs will be generated to visualize performance differences as the number of drones increases.
    - We will average performance metrics over multiple trials to ensure statistical accuracy.
- **Analysis:**
    - We plan to highlight key differences in how each approach handles larger swarm sizes, focusing on potential trade-offs between development simplicity, execution efficiency, and memory consumption.
    - We will discuss the pros and cons of custom development versus using existing libraries for machine learning-driven swarm behaviors.

5. Conclusion

This project provided valuable insights into the advantages and limitations of simulating drone swarm behaviour using two different approaches: a from-scratch method and a library-based solution. The from-scratch approach allowed for greater customization and control over the swarm algorithms, such as Boids, but at the cost of increased development complexity and lower scalability when handling larger drone swarms. On the other hand, the library-based approach, particularly using Unity ML-Agents, offered a more streamlined implementation, reducing the development time while leveraging reinforcement learning to achieve more sophisticated swarm behaviours.

From a performance perspective, the from-scratch approach generally performed better in terms of execution time and memory usage for small to medium-sized swarms, while the library-based approach exhibited more efficient scalability as the swarm size grew, albeit with higher initial memory overhead. In conclusion, the choice between these approaches depends on project requirements, with the from-scratch method being ideal for lightweight, custom simulations, while the library-based approach excels in scalability and ease of development for complex simulations.