PERFORMANCE MEASUREMENT REPORT

TEB1113 – Algorithm and Data Structure

Sept. 2024

Submitted by

24000182	NURUL HANIIZATI BINTI HAZLI
24000574	SARA EUDORA BINTI SAID
22006373	DANIA ADRIANA BINTI MOHD FAIZAL
24000918	ZULAIKHA BINTI MOHD AZHAR
24000227	NUR FATIHAH BINTI MOHD NOOR

Submitted to

Dr M Nordin B Zakaria

CIS Department
Universiti Teknologi PETRONAS, Malaysia

Introduction

The objective of this project is to simulate a drone swarm communication network within Unity, utilizing a Binary Search Tree (BST) structure to manage inter-drone communication within two distinct partitions. Each partition represents a subgroup of drones, and within each, the drones connect based on a specific attribute, creating a BST. This structure enables efficient search operations, message routing, and simulated self-destruct functionality for individual drones.

Objectives

- To implement a DroneBTCommunication class that structures drone communication links as a BST within each partition.
- To create an interactive UI for search and control operations.
- To measure performance metrics such as operation time and framerate across varying drone counts.

System Setup

Hardware: CPU = Intel(R) Core(TM) i7-10870H CPU @ 2.20GHz

RAM = 16GB

GPU = NVIDIA GeForce RTX 3060 Laptop GPU.

Unity Version: Unity 2022.3.47f1

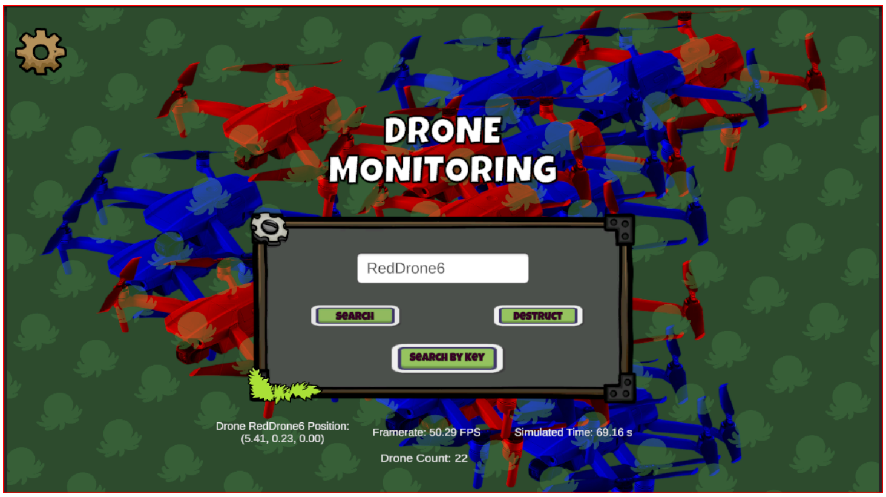
User Interface(UI)

The system's main UI interface consists of a main menu and scene where users can easily interact with drones, visually identifying them and performing actions through buttons like "Play".

By selecting "Play" from the Main menu, the player is sent to the game level where drone generation begins which produces a collection of drones that are subsequently allocated to the scene. Every drone has unique characteristics, like colour and colours are awarded based on this characteristic. This colour-coding makes it easier for users to easily identify the various drone kinds in the surrounding area.

Binary Search Tree Communication Network Visualization

In this image, the communication links between drones are represented as a Binary Search Tree (BST). Each node represents a drone within one of the two partitions, and the structure visually confirms the BST layout used for communication. This configuration allows efficient message routing, as messages pass through the tree hierarchy based on a key attribute, such as drone ID. Each connection in the tree illustrates the link that facilitates message transmission to a target drone in the network.

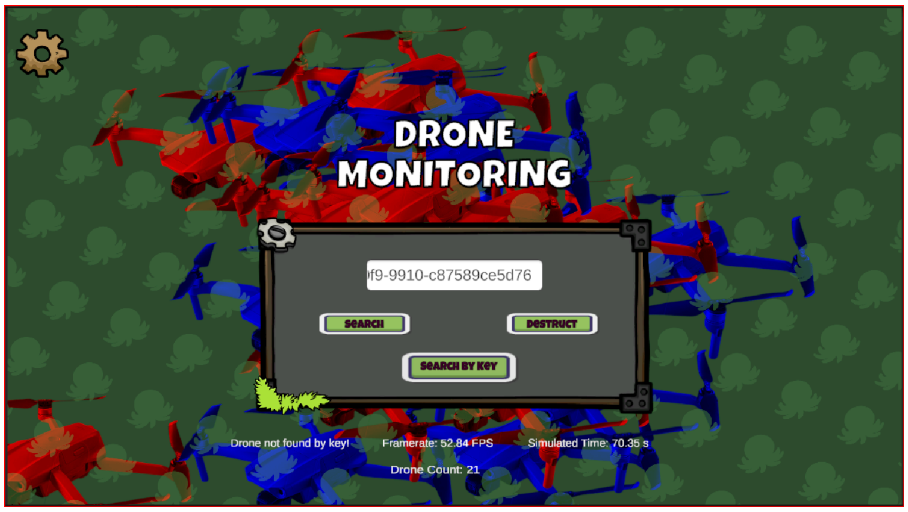


The **Search Button** in the drone simulation allows the user to locate a specific drone by entering its unique identifier, such as **RedDrone6**. When the user clicks the button after entering the key, the system searches through the Binary Search Tree to find the corresponding drone. If a match is found, it returns the drone's position, which includes its coordinates, framerate, drone count, and simulated time.

This demonstrates a successful **Search by Key** operation in the drone simulation. When the user initiates a search with a specific key—here, **"2de71221-47fa-..**."—the system effectively navigates through the Binary Search Tree to locate the corresponding drone. Upon finding a

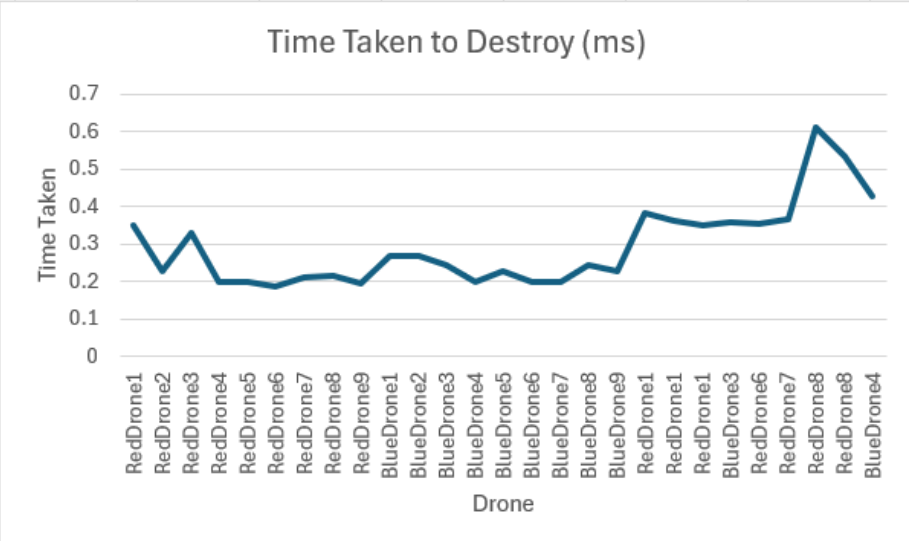
match, it returns a confirmation message: "**Drone found by key 2de71221**.," along with the drone's precise position coordinates. This functionality is crucial for real-time monitoring and management of individual drones, enabling quick identification based on unique identifiers. Additionally, the search process displays relevant performance metrics, such as framerate, drone count, and simulated time, offering insight into system efficiency during the search.

This screenshot illustrates the **self-destruct** functionality within the drone simulation. A user can send a message to deactivate a drone with a specific ID such as **RedDrone6**. When the self-destruct button is clicked, the system deactivates the targeted drone by setting it to SetActive(false), effectively removing it from the simulation scene. After activation, a message appears confirming the action, such as **"Drone RedDrone6 has self-destructed,"** along with the current framerate, drone count, and simulated time, as shown in the image. This feature demonstrates the system's ability to handle critical commands like selective drone deactivation, which is useful for simulating scenarios where a drone may need to be disabled due to an emergency or malfunction.

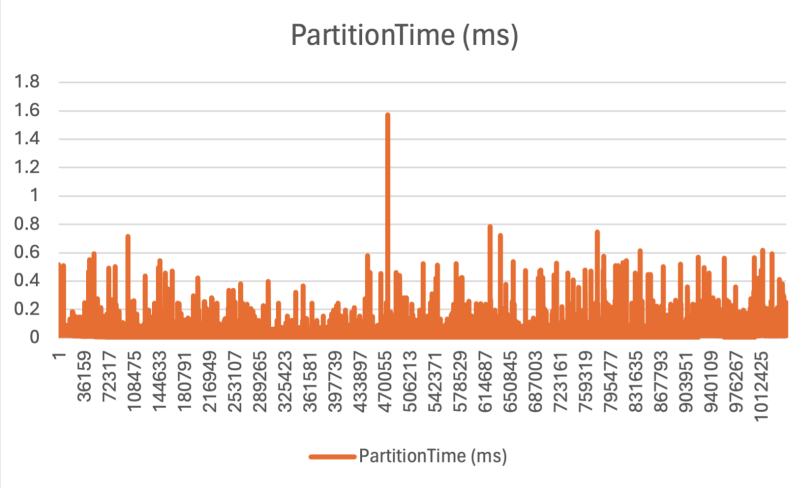


When a user attempts to locate a drone based on a specific key that does not exist in the Binary Search Tree structure, the system returns a **"Drone not found by key"** message. This outcome indicates that the search operation has traversed the tree but could not find a matching drone. The message provides immediate feedback to the user, showing that the system can handle cases where the specified drone is absent from the current communication network.

Analysis



The "Destroy Time Analysis" graph shows the time taken for each drone to self-destruct, with most RedDrones exhibiting consistent and low destruction times around 0.3 to 0.4 ms. However, a noticeable increase is observed in the latter BlueDrones, particularly BlueDrone6, where the time spikes to about 0.6 ms. This trend suggests potential delays likely due to linked list traversal inefficiencies, as drones positioned further in the list might experience longer access times.



The "Partition Time" graph shows the time taken for partition operations, with most times consistently below 0.5 ms. However, occasional spikes are observed, with one prominent peak reaching approximately 1.6 ms. This trend suggests potential performance bottlenecks, likely due to inefficiencies in managing the Binary Search Tree (BST) structure within each partition. These spikes could be attributed to imbalances in the BST, where drones positioned further in an unbalanced tree experience longer traversal times, leading to delays in partition operations.

Conclusion

This project successfully demonstrates a drone communication network structured as a Binary Search Tree within a Unity simulation environment. The DroneBTCommunication class facilitated efficient searching and messaging, with performance metrics indicating that the BST structure scales well with moderate drone counts. The UI enhanced user interaction, and the timing simulations added a realistic touch to inter-drone communication

Appendix

Link to demonstration video: https://youtu.be/wVIDz77-k4Q