Team 7

MathWorks

UAV Path Planning

Juan Tiguila
Jason Alvarez
Jonathan Dang
Jade de Jesus
Abraham Diaz
Marcos Olvera
Bryan Segovia
Prashant Tewary
Kevin Velez
Erick Vergara

Advised by Dr. Manveen Kaur

Overview

1. Project Scope
2. Problem Components
3. Demo
4. Future Goals

Motivation

Revolutionizing urban transportation with urban air mobility

Orchestrating collision-free paths for multiple drones operating within the same environment

Opportunity to make the future of transportation and logistics efficient and sustainable

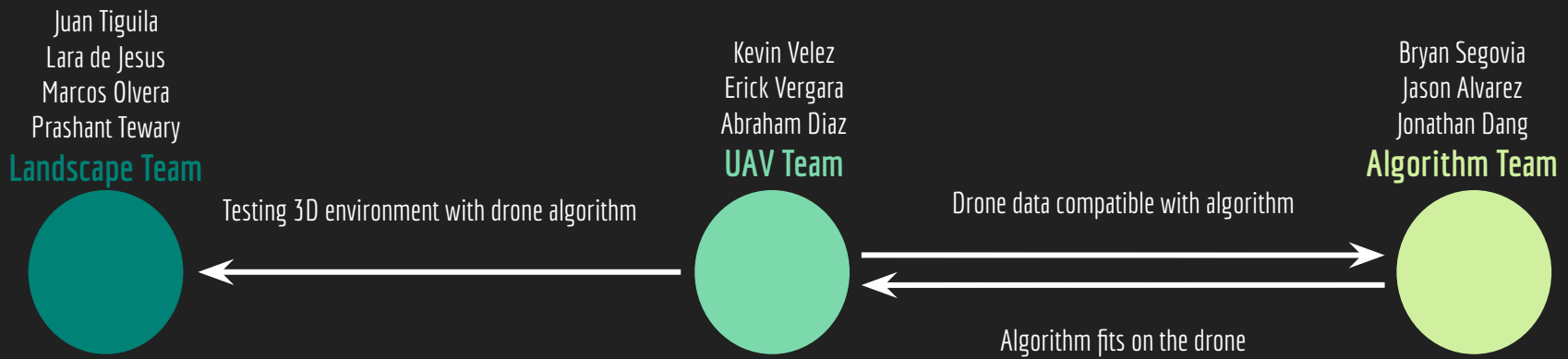
Problem Statement

The rise of urban air mobility, powered by UAVs, can transform urban transportation efficiency and sustainability; however, the simultaneous deployment of multiple drones in shared urban spaces requires the development of a collision-free path planning algorithm.

Scope

1. Skill Enhancement
2. Scenario Simulation
3. Single-Drone Path Planning
4. Multi-Drone Coordination
5. Centralized Tracking
6. Performance Evaluation

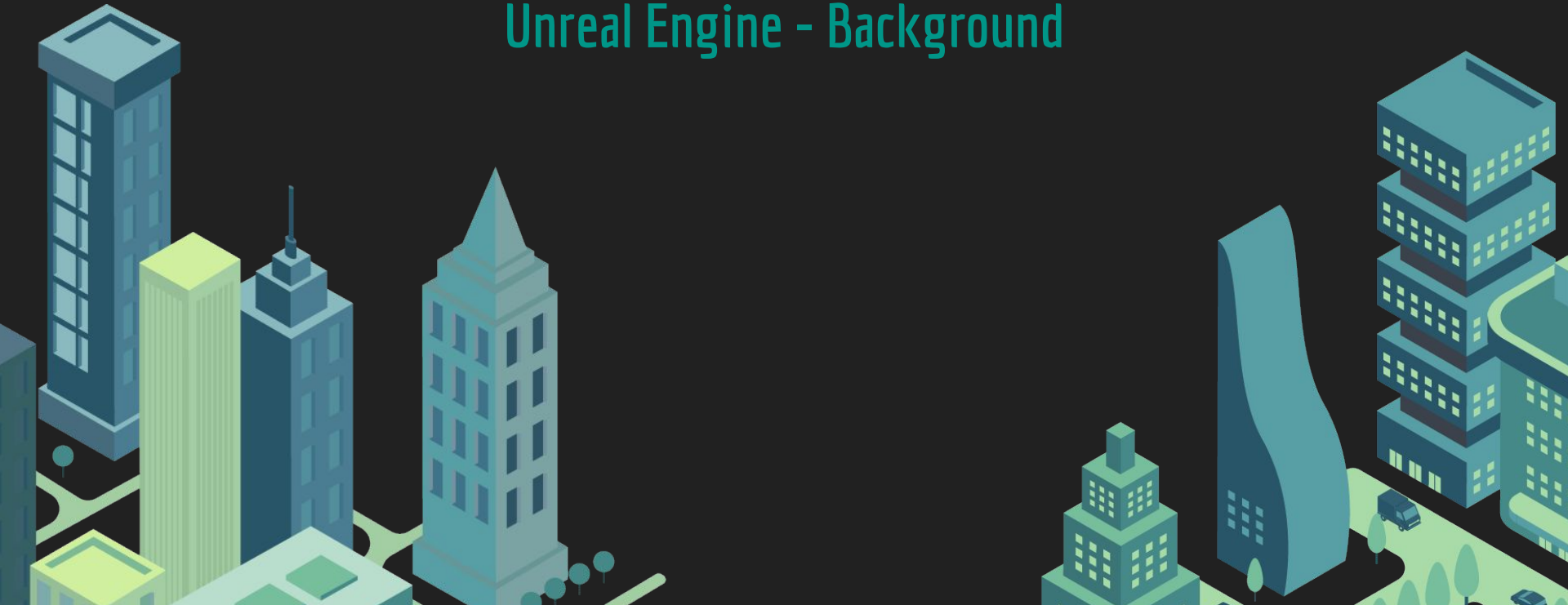
Team Workflow Chart



Team Lead: Jade de Jesus
Communications Lead: Juan Tiguila
Documentation Lead: Abraham Diaz

Landscape

Unreal Engine - Background



C++ in Unreal Engine

- Ideal for performance-critical tasks, ensuring optimal execution speed and resource utilization.
- Used to implement artificial intelligence (AI) behaviors for non-player characters.

Blueprints in Unreal Engine

- It is a visual scripting system that allows developers to create and manipulate game logic, behaviors, and assets.
- Rapid Prototyping

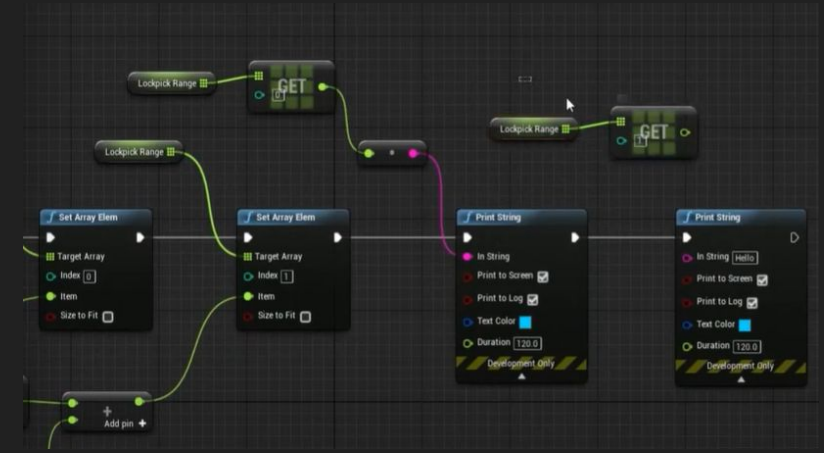


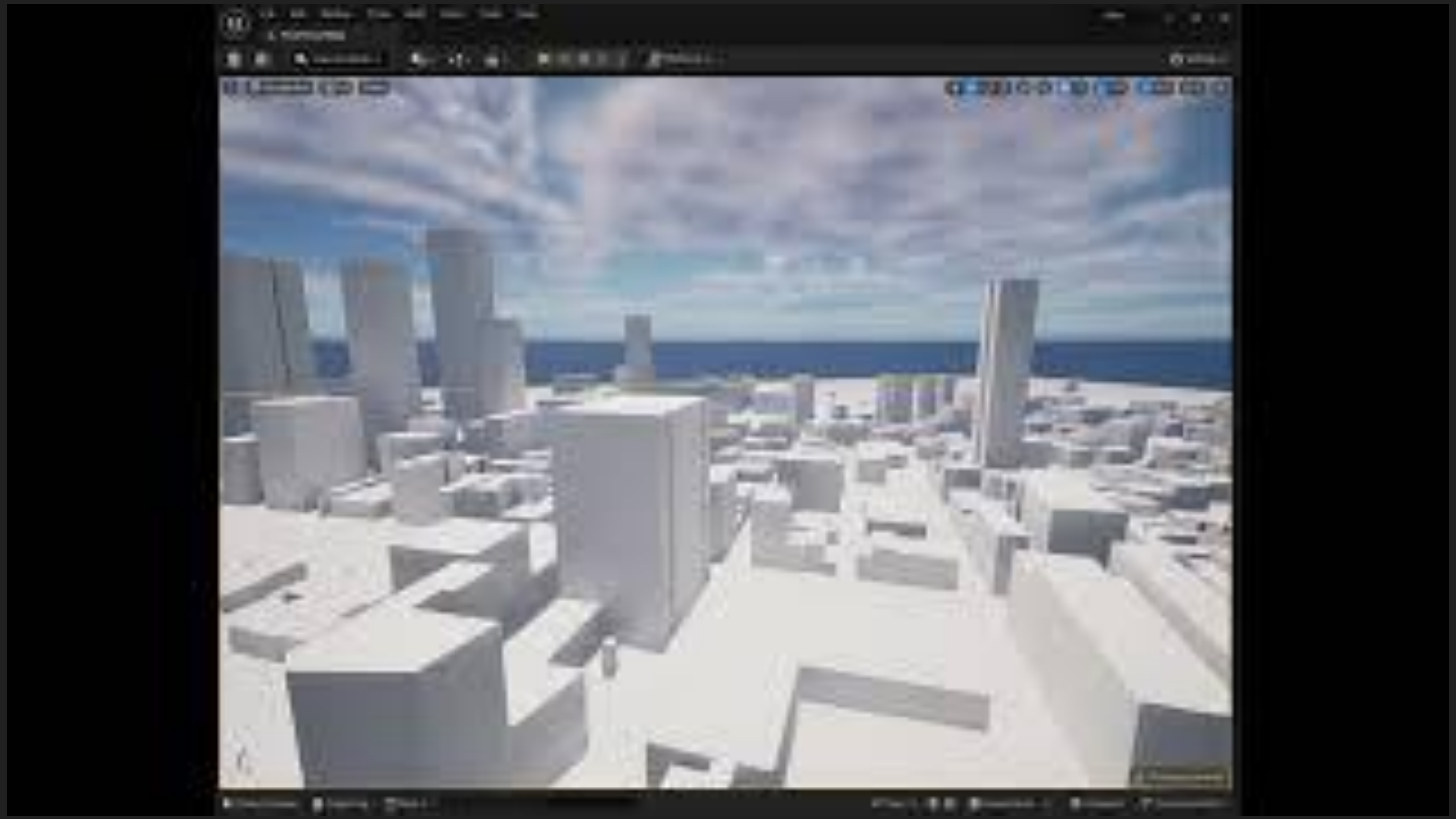
Figure 2: Unreal Engine Blueprints

C++ vs. Blueprints in Unreal Engine

- C++ is more broad and offers more functionalities than Blueprints.
- C++ is used for optimizations to write more efficient algorithms or low-level code.
- C++ can be integrated with Blueprints, a hybrid approach where performance-critical code is implemented in C++, and higher-level logic is handled in Blueprints.
- Blueprint is a visual scripting language which uses graphical nodes
- Blueprint is entry level program but also has more powerful tools for advance programing

Youtube: Nitrogen at https://www.youtube.com/watch?v=ueOqcwlFfMc

Landscape - Scene



Algorithm

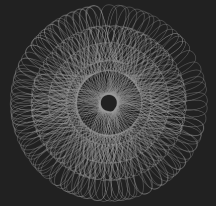
Path-Planning Development

Algorithm

A* is designed to find the shortest path from a starting point to a goal point in a graph or grid-based environment.

- It combines the advantages of Dijkstra's algorithm with heuristic search
- Guarantees finding the shortest path.
- Employs a heuristic to guide the search, reducing computational requirements.

Algorithm



PROS

Optimality: A* is known for finding the optimal path, which is crucial in urban environments where efficiency and safety are paramount.

Efficiency: A* can be more efficient than other algorithms, such as Dijkstra's, for finding the optimal path. It uses heuristics to guide the search, which can significantly reduce the search space.

Real-Time Capabilities: A* is capable of real-time path planning, making it suitable for dynamic and complex urban environments where paths may need frequent updates.

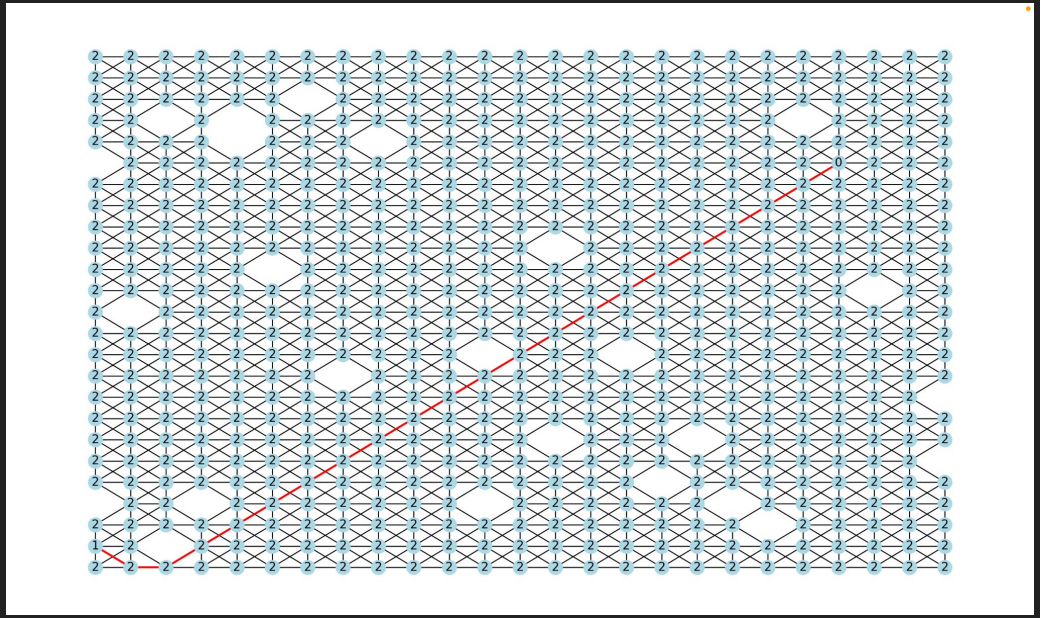
Global and Local Planning: A* can be used for both global path planning (long-range navigation) and local path planning (short-range obstacle avoidance).

CONS

Memory Usage: A* can be memory-intensive, particularly in large environments, as it needs to store information about the nodes it has explored. This can be a limitation in resource-constrained environments.

Complexity of Implementation: While not as complex as some other algorithms, A* still requires careful implementation, including handling open and closed sets, and choosing an appropriate data structure.

Algorithm



- Grid of nodes
- Objects are removed from the grid to make sure that we don't fly through them
- Red line represents the path the drone will take.
- Below is the array of coordinates that the drone will take.

[(0, 1), (1, 0), (2, 0), (3, 1), (4, 2), (5, 3), (6, 4), (7, 5), (8, 6), (9, 7), (10, 8), (11, 9), (12, 10), (13, 11), (14, 12), (15, 13), (16, 14), (17, 15), (18, 16), (19, 17), (20, 18), (21, 19)]

UAV

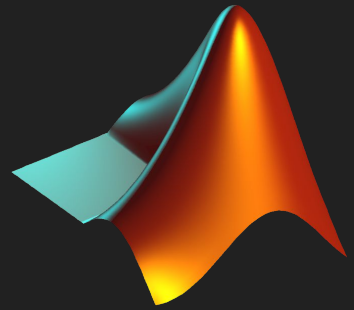
Drone Development

UAV

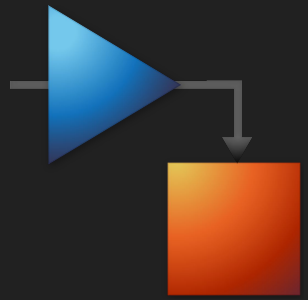
Objective: Specify the baseline and if possible minimal hardware limitations of the drone that meet the fractional computation time requirements for the underlying algorithm to meet and providing a controllable drone to simulate the automated algorithm in a urban landscape.

Matlab and Simulink

Matrix efficient programming language



Graphical programming language



The UAV Drone

The MatLab quadcopter project illustrated an example UAV drone with all of its basic required inputs for path planning.

UAV

Will briefly describe the baseline hardware and specify the ***main hardware***

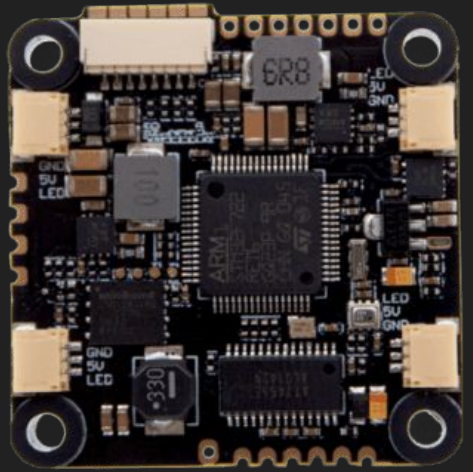
- Pixhawk PX4
- Bosch Sensortec BMI088
- Raspberry PI

Flight Controller

The flight controller is the brain of the drone

Monitors and controls every action the drone does

- Balance
- Stability
- Orientation



Pixhawk PX4 PIX 2.4.8

Great for multicopters

Integrated backup power supply

Built in gyroscope, accelerometer, magnetometer and barometer

~$169.99

IMU Functionality

- Collects sensor data on motion, orientation, and sometimes magnetic fields.
- Processes and fuses sensor data to determine:
  - Orientation in three-dimensional space
  - Changes in velocity, position, and altitude
- Provides real-time information for control systems

Bosch Sensortec BMI088

- The Bosch Sensortec BMI088 is a high-performance Inertial Measurement Unit (IMU) designed for precise motion sensing in various applications.
- Integrates a 3-axis accelerometer and a 3-axis gyroscope.
- Offers precise measurement capabilities for motion tracking and orientation sensing.

- Advantages
  - Offers consistent and reliable performance in motion sensing applications.
  - Optimized power usage for extended operation, ideal for battery-powered devices.
  - Enables energy-efficient performance without compromising accuracy.

Microcontroller

The microcontroller runs a single program repeatedly

Can run programs with more than one process

Firmware for communication with other running software on the drone

Talks to the flight controller about the environment

- Mission planning and navigation
- System health
- Emergency signals
- Communication with companion computers

Raspberry Pi

PROS

Suitable for various applications beyond drone control
- *image processing, computer vision, and communication.*

Can run Linux and support a wide variety of programming languages like C++ and Python.
- This means the drone can be interfaceable

Lots of computation power compared to microcontrollers

Attachable hardware
- Camera port, ethernet & wireless connectivity, USBs HDMI monitor, audio jack, GPIO, HATs, heat sinks, modules

CONS

Has overhead which impacts responsiveness
- Not real-time considering the importance of stabilization and control algorithms for drones

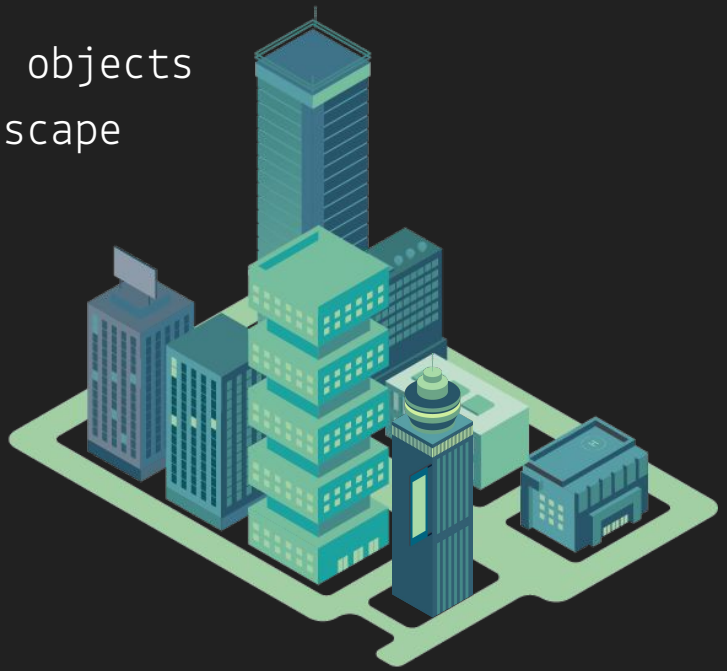
May notice fractional increase in delays with Python

High power consumption
- Simultaneously runs an Linux operating system

- $189.99 mini board computer
- 1.5GHz clock speed at 5V

Project Goal and Future Work/Plan

- Allow a single drone to navigate a three-dimensional simulated environment
- Have a drone detect static and moving objects
- Introduce multiple drones in the landscape



Conclusion

To carry out this project we split into 3 groups that focused on one major aspect of the project

- Landscape: created a 3D landscape to test the drone and its algorithm
- Algorithm: made to detect objects and maneuver through the landscape
- UAV: uses the algorithm and test its capabilities/functionality

Integration of MatLab and UE

- MatLab used to create algorithm and coding aspects
- Unreal Engine used for landscape design and testing the UAV

Resources and References

- Link to our GitHub repository
- Software Requirements Specification
- Software Design Document
- https://www.youtube.com/watch?v=ueOqcwlFfMc

Thank you!