FAMU-FSU College of Engineering
Department of Mechanical, Electrical, and Computer Engineering

Team 315

Team Members: Colby Hackett, Jimmy Lu, Cameron Sayers, Morgan Skinner, Jackie Ou, Jonathan Tooby

Date: 10/4/24

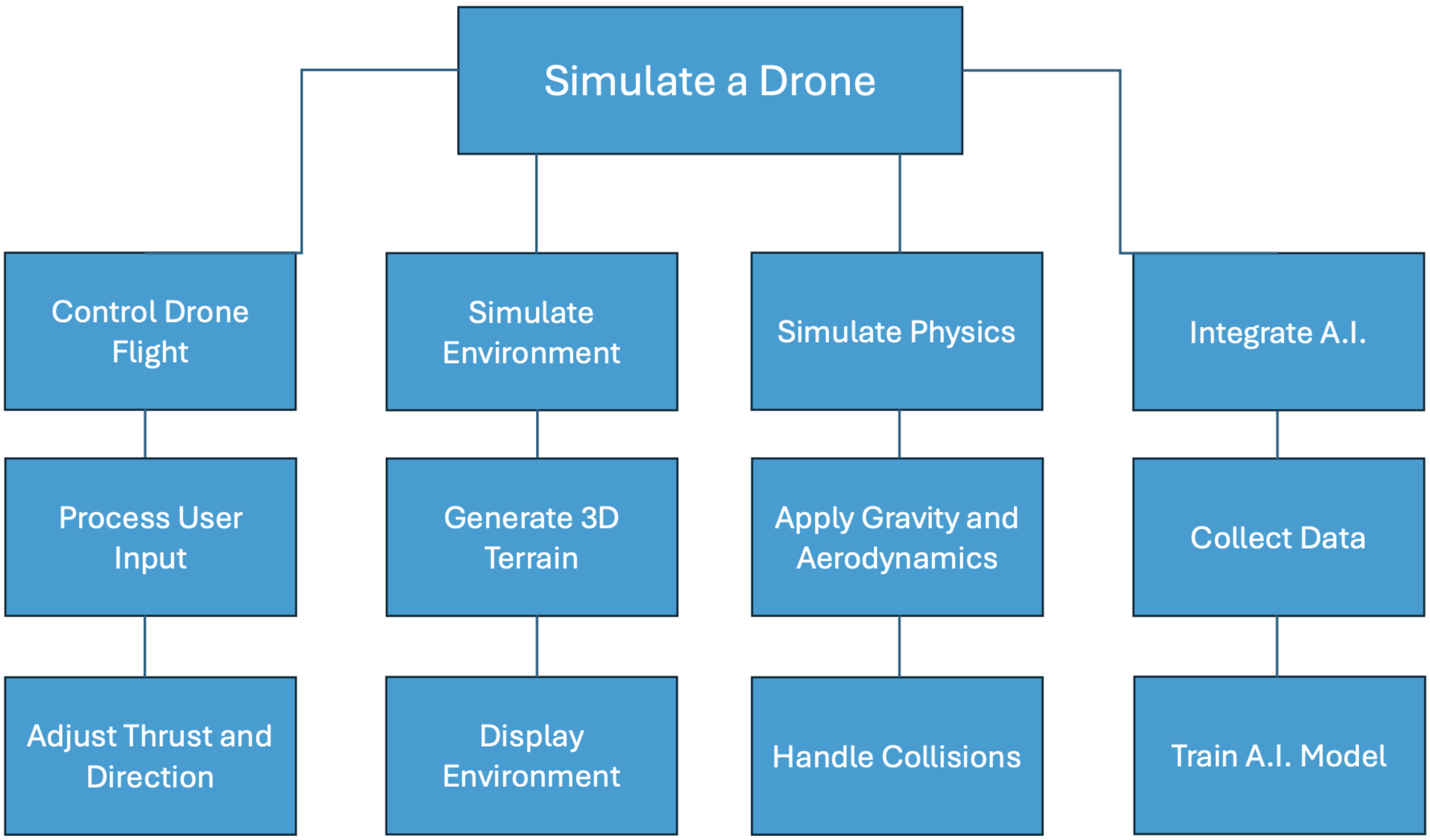
Functional Decomposition

Tables of Contents

Introduction

The purpose of this document is to provide an in-depth functional analysis of the systems involved in our senior design project: a drone flight simulator with AI integration. By methodically decomposing the project into its core functions and sub-functions, we aim to gain a detailed understanding of the technical requirements, dependencies, and interrelations of each system component. This structured breakdown will clarify the project's overall scope and define the specific actions required for successful completion. To achieve this, the document is organized into several key sections, including a function tree, which visually represents the hierarchical structure of the system's functions, and a cross-reference table that outlines the interaction between various functions and project requirements. Additionally, the decomposition levels will guide us through the progressively detailed tasks within each major system, ensuring that every aspect of the project is accounted for and planned methodically. This analysis serves as a roadmap for our team, providing a clear vision and structured plan to guide us through each phase of development. With this comprehensive approach, we ensure that every function and sub-function is fully understood, enabling us to deliver a robust, well-executed final product.

Function Tree



Cross Reference Table

	System				
Function	Control Drone flight	Simulate environment	Simulate physics	Integrate AI	Needs
Process user input	X				6
Adjust thrust and direction	X				6
Generate 3D terrain		X			2,5
Display environment		X			2,5
Apply gravity and aerodynamics			X		1
Handle collisions			X		1
Collect data				X	4
Process data				X	4

Need 3 is making the whole system portable, which is not a function of the actual simulation itself.

Decomposition levels

1. Control drone flight
    a. Process user input from controller or keyboard
        i. Translates user input into realistic drone movements in the simulation
        ii. Provides feedback (vibration) based on simulation circumstances
    b. Thrust and Direction Control
        i. Adjust throttle (altitude control)
        ii. Adjust pitch, roll, and yaw (orientation control)
2. Simulate environment
    a. Terrain Generation
        i. Create 3D objects (terrain, obstacles)
        ii. Use Rigidbody planes for environmental physics
3. Simulate Physics
    a. Gravity and Aerodynamics
        i. Apply gravity to the drone (weight)
        ii. Simulate drag, lift, and other aerodynamic forces
    b. Collision Handling
        i. Use Box Collider and Rigidbody components (Unity-based physics)
4. Integrate AI
    a. Data Collection
        i. Capture flight and user data
    b. Process data
        i. Use data to train ML or AI model

Description of modules

Level 1

Module	Drone flight control
Input	User input from controller or keyboard
Output	Drone movement (thrust, yaw, pitch, roll) in the simulation
Functionality	Translates user input into realistic drone movements, controlling thrust, yaw, pitch, and roll to simulate flight.

Module	Simulate Environment
Input	Simulation settings and prebuilt 3D objects
Output	3D terrain and environment visuals (obstacles, scenery)
Functionality	Generates the 3D environment, including terrain and obstacles, for the drone to interact with, creating a dynamic simulation space.

Module	Simulate Physics
Input	Drone position, velocity, environmental factors
Output	Gravity, aerodynamics, and collision effects on the drone
Functionality	Simulates realistic physical interactions such as gravity, aerodynamics, and collisions, adjusting the drone's movement based on environmental forces

Module	Integrate AI
Input	Flight data and user input
Output	AI-driven decisions
Functionality	Uses flight and user data to inform or train AI algorithms

Module	Process User Input
Input	Wired or wireless connection to simulation
Output	Drone movement signals (direction, throttle)
Functionality	Processes user inputs and translates them into movement signals that control the drone’s thrust, yaw, pitch, and roll in the simulation.

Level 2

Module	Thrust and Direction Control
Input	Processed user input (throttle, direction)
Output	Drone altitude (throttle) and orientation (yaw, pitch, roll)
Functionality	Controls the drone's thrust (altitude) and direction (yaw, pitch, roll), enabling smooth and precise movements in the simulation.

Module	Terrain Generation
Input	Terrain parameters and prebuilt unity assets
Output	3D terrain and obstacle models
Functionality	Generates the simulation environment using 3D objects, including terrain and obstacles, for the drone to navigate.

Module	Rigidbody Planes
Input	Unity’s Physics settings for terrain
Output	Physics-enabled environment objects
Functionality	Applies Unity’s Rigidbody physics to environment objects for realistic interactions with the drone.

Module	Gravity and Aerodynamics Simulation
Input	Drone mass, velocity, and environmental conditions
Output	Simulated forces (gravity, drag, lift) on the drone
Functionality	Simulates gravity and aerodynamic effects on the drone, including weight and drag forces during flight.

Module	Collision Handling
Input	Drone's position and velocity, surrounding objects
Output	Collision responses
Functionality	Detects collisions between the drone and obstacles, applying Unity’s Box Collider and Rigidbody to handle the resulting impact.

Module	Data Collection
Input	Flight data and user inputs
Output	Usable data for the AI algorithm
Functionality	Captures data from the drone’s flight or user inputs for analysis or AI model training.

Module	Data Processing for AI
Input	Captured flight and user data
Output	AI-driven outputs or model training
Functionality	Processes flight data for machine learning models, improving AI-driven decision-making for future flights.

Summary

Breaking down a project into different functions is helpful in planning the final product as a whole. It allows the team to create various sections of the project and test their functions before moving on to another system that relies on the previously implemented system. This modular approach ensures that each component works independently and reduces the risk of errors cascading through the entire project. By isolating subsystems, debugging becomes more manageable, and individual team members can work in parallel, speeding up development. Additionally, it provides flexibility, enabling easier updates or changes to specific parts without overhauling the entire system. In the context of larger projects, such as a VR drone, this method enhances scalability and simplifies the integration of advanced features as the project evolves.