

ECE 445
SENIOR DESIGN LABORATORY
DESIGN DOCUMENT

Campus Tour Guide by AI-Powered Autonomous System

Team #21

XUANBO JIN
(xuanboj2@illinois.edu)

HAO REN
(haor2@illinois.edu)

YUNTONG GU
(yuntong7@illinois.edu)

WEIANG WANG
(weiangw2@illinois.edu)

TA: Xinlong Huang

March, 18, 2024

Contents

1	Introduction	1
1.1	Problem and Solution Overview	1
1.2	Visual Aid	1
1.3	High-level requirements list	1
2	Design	3
2.1	Block Diagram	3
2.2	External Subsystem Motion Control Subsystem (UAV)	3
2.3	Sub-system Overview	3
2.4	AI-powered response generation system	4
2.4.1	System Architecture and Design	4
2.4.2	Input and Output of the Subsystem	5
2.4.3	Multi-media Data Collection	6
2.4.4	Intent Identification and Retrieval	7
2.4.5	Protection Sub-Unit	7
2.4.6	Goals and Verification	7
2.4.7	Version Iteration	8
2.4.8	AI-powered Response Generation Subsystem Interface	9
2.5	User Interface	10
2.5.1	Subsystem Architecture and Design	10
2.5.2	Input and Output of the Subsystem	11
2.5.3	Frontend Development	12
2.5.4	Remote Server Setup	12
2.5.5	Connection with AI Subsystem	12
2.5.6	Verification and Results	13
2.6	Planning & Control Subsystem	15
2.6.1	Notation and Explanation	15
2.6.2	Subsystem Architecture and Design	16
2.6.3	Input and Output of the Subsystem	16
2.6.4	Data Collection and Tagging	17
2.6.5	Algorithms for Subsystem	17
2.6.6	Integration with QGroundControl Platform	18
2.6.7	Verification and Results	19
2.7	Sensor System	21
2.7.1	Introduction	21
2.7.2	Temperature and Humidity Sensor	21
2.7.3	Accelerometer Sensor	22
2.7.4	LCD Display	23
2.7.5	PCB design	24
2.7.6	Verification and Requirement Plan	24
2.8	Tolerance Analysis	26
2.8.1	Route Planning Stability	26
2.8.2	Communication System Delay	26

2.8.3	GPS Locating Error	27
3	Cost and schedule	29
3.1	Cost Analysis	29
3.1.1	Labor	29
3.1.2	Parts	29
3.1.3	Grand Total	30
3.2	Schedule & Milestone	30
3.2.1	Milestone 1: Basic Hardware Configuration and Functionality	30
3.2.2	Milestone 2: Software Integration and Functional Testing	31
3.2.3	Milestone 3: Basic Autonomy and Obstacle Awareness	31
3.2.4	Milestone 4: Advanced Autonomy with Obstacle Navigation	32
3.2.5	Milestone 5 [Optional]: Interactive Communication and Control	32
3.3	Weekly schedule	33
4	Ethics and Safty	36
4.1	Ethical Considerations	36
4.2	Safety Considerations	36
	References	37
	Appendix A Requirement & Verification Table	38

1 Introduction

1.1 Problem and Solution Overview

Anyone entering a place for the first time, like an university, can be quite challenging. Knowing where you are, how to get to your destination, how to optimize your routes, knowing factors that will influence your routes can be complicated. Having a real-time interactive system that guides people through this process is needed. It has been possible yet not able to scale because it's not open-sourced, and its hardware isn't standardized, and is expensive. The interaction isn't versatile enough to adapt well under the ever-changing applications. A cheap and versatile solution is needed.

Our solution utilizes autonomous UAV to guide our clients, sensing them and the environment, such as obstacles and drone's location with a sensor module, controlled by a control unit which orchestrate a series of tasks. Our solution is cheap, open-sourced, and versatile to meet the need of a generalized and sustainable long-term solution for our campus and many other applications.

1.2 Visual Aid

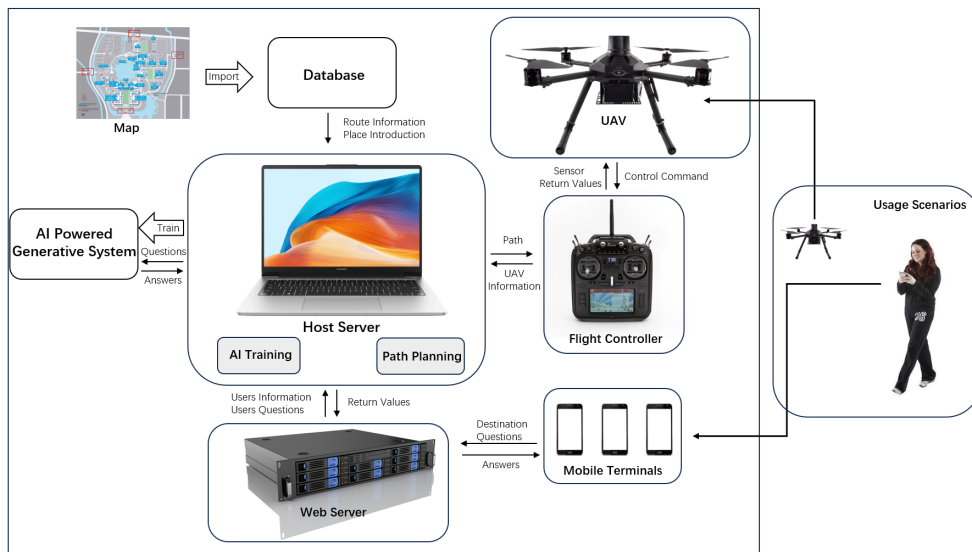


Figure 1: Visual Aid

1.3 High-level requirements list

- There should be a UAV that can fly autonomously from a preset starting point to a preset destination, and has enough endurance to achieve our requirements
- There should be a complete user UI interface for interacting with the user and displaying the information the user needs in real time

2 Design

2.1 Block Diagram

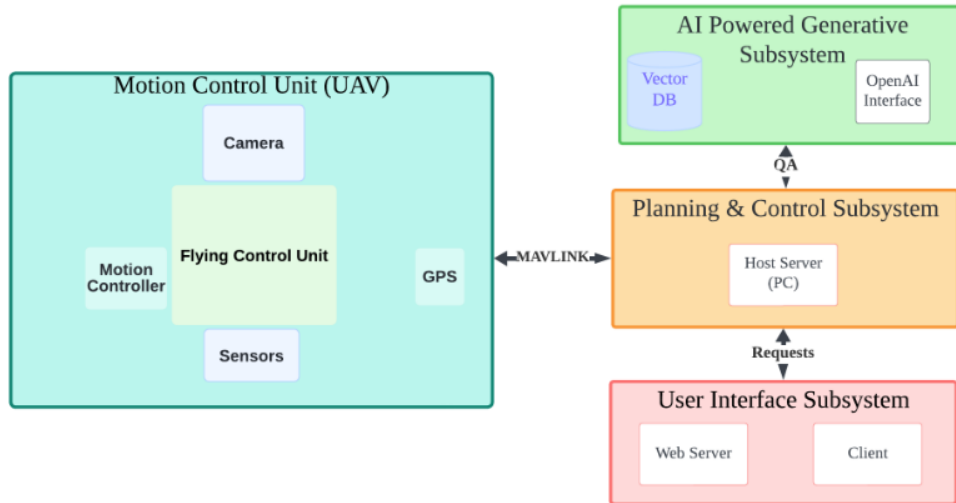


Figure 3: Block Diagram

2.2 External Subsystem Motion Control Subsystem (UAV)

The UAV model used in our project is the MFP450, a medium-sized drone platform with a 410mm wheelbase. It is equipped with a Pixhawk 6C open-source flight controller, M8N-GPS, brushless motors, custom hard-shell batteries, Minihomer telemetry, an integrated optical flow ranging module, camera, and other devices. This UAV meets the requirements for stable flight both indoors and outdoors, and it is suitable for various applications including teaching and development. This is an off-the-shelf open source UAV, so we won't go in depth here.

2.3 Sub-system Overview

This project consists of 4 subsystems:

- AI-powered response generation system
- User Interface
- Planning & Control system
- Sensor System

These sub-systems sense the context of the tour, planning the tour, and interact with users in a natural manner. The emphasis of the sub-systems are the integration of strong ability of Retrieval-Augment Generation into a mobile system. **These subsystems link the**

cloud, the PC, and the embedded mobile systems into an interactive tour guiding assistant.

2.4 AI-powered response generation system

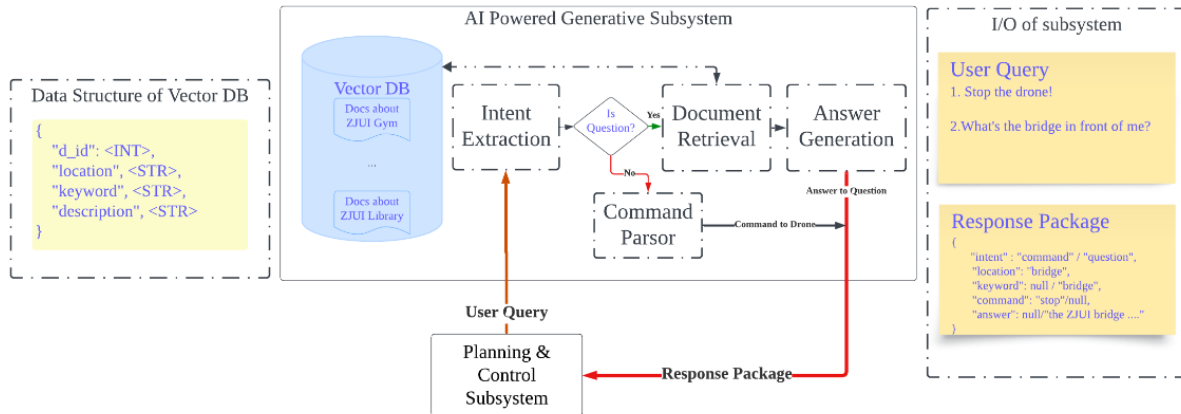


Figure 4: AI-powered response generation system

This model focuses on building an assistant to answer user queries about the ZJU-UIUC campus. The user's query can be a question about the campus or a command to the drone. The text generation and embedding modules are powered by OpenAI (OpenAI, 2023) and are hosted in their clouds.

2.4.1 System Architecture and Design

The system is architected to efficiently handle two distinct types of queries: informational and operational. The architecture is modular, with each module specializing in tasks such as intent detection, data retrieval, and command validation. This modular design facilitates scalability and maintenance while ensuring the system can evolve to incorporate future enhancements or functionalities.

The system can be broken into the following parts:

- Vector DB storing related campus material
- Intent Extraction module extracting user's intent.
- Search Engine module search for related entries in Vector DB
- Answer Generation

To build this system, we design the following steps:

- Collect related data and tag them into a vector database.
- Design a simple framework, given a user question + context, output an answer.

The **workflow** is as follows: Before the workflow, we first store the document of the campus in a vector DB in a well-defined data structure.

Branch1: When the user’s input is a question to the ZJU-UIUC campus.

1. Upon each user input, we detect the intent of the user. If the intent is asking questions,
2. We retrieve the document related to the user’s question in the vector database.
3. Combined with the retrieved info and the prompting, we output the answer to the user’s question.

Branch2: When the user’s input is a command to the UAV.

1. Upon each user input, we detect the intent of the user. If the intent is commanding the drone,
2. We select one of the valid commands and then pass it to the command-validator.
3. If it’s a valid command, we pass it to the drone.

2.4.2 Input and Output of the Subsystem

Table 1: Input and Output of the AI-Powered Response Generation Subsystem

Field Name	Type	Meaning
User Query	Input	Campus related query or a command to drone
User Location	Input	Current GPS location of user
Answer	Output	Answer to user’s question
Command	Output	Parsed output Command to the drone

2.4.3 Multi-media Data Collection

The GPT model does not have the ability to answer questions related to ZJU-UIUC. It failed to answer the 100 testing questions completely. To supply pertinent information regarding to users' query, we need a set of data tailored for our application. The data we will use are in multiple forms as shown, it could be digital pdfs, or paper-based materials placed at many places at different locations within the campus. We utilize several methods to handle these different forms of information.

1. OCR tool backed by Wechat.
2. python tools to parse pdfs.

The code to parsing multi-media material is in this directory.

We convert these data-sources into uniform textual information so our agent can answer user's query with accurate and diverse information.

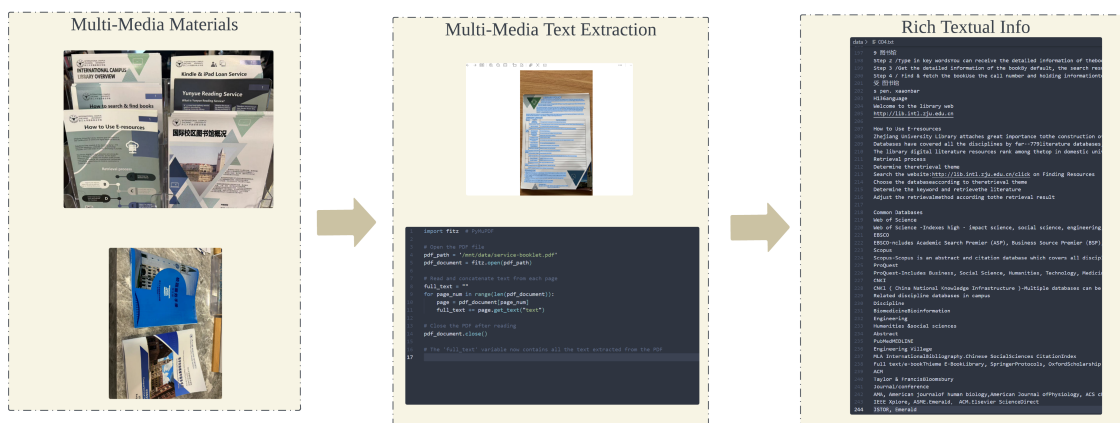


Figure 5: Workflow of Data Collection

Table 2: Vector DB Data Store Table

Field Name	Type	Meaning
d_id	INTEGER	Indexing of the entries
location	STRING	Location of the described place
outlook	STRING	Appearance of the location
keyword	STRING	keyword of the description
description	STRING	a text describing related info about a location in ZJUI campus

2.4.4 Intent Identification and Retrieval

The intent identification module classifies users' input into three categories, as specified in the instruction section of the prompt. This module relies on a straightforward query to GPT 3.5. However, given that GPT models typically achieve only about 70% accuracy on general multiple-choice tests [1], we must evaluate its performance on our specific task with caution. Enhancements might be achieved through refined prompt engineering [2].

The information retrieval module employs a Retrieval-Augmented-Generation approach. After extensive review [3], I opted for a sophisticated modular configuration. This agent integrates multimedia input processing, a search engine as the retriever, and a generation unit that processes and outputs the retrieved information.

The document retriever connects queries to relevant locations and retrieves the associated data. However, as data for each location increases, it may surpass GPT's token limit—the maximum number of tokens an LLM can process at one time. To manage this, the retriever breaks down data into manageable "chunks," summarizing each chunk and converting them into vectors using an OpenAI tool. The vector distance indicates the textual semantic similarity, which aids in identifying the most relevant chunks for a query.

2.4.5 Protection Sub-Unit

Just like any hardware system, the subsystem's IO and behavior needs appropriate amount of protection. For instance, we must ground the I/O of the module so that the module would not output illegal commands or commands that exceed the tolerance of the other units.

The specific groundings are listed in the following table:

Table 3: AI-Powered Response Generation Subsystem Protection Unit

Index	Protection Object	Protection Explanation
0	Input text	Either campus-related question or a command to drone
1	Output Text	Must have a length between 0 to 255 words
2	Output Command	Valid command to drone
3	Output Text	Must exceed confidence threshold

2.4.6 Goals and Verification

In order to verify the design, our system has the following 3 main target functionalities:

- Identify the intent of the input to the subsystem

- Fetch the correct external materials
- Output proper answer

In order to verify these functionalities, we design the following verification methods:

- The intent is evaluated automatically against the ground truth labeled by human on a testing dataset containing 100 sample questions.
- The retrieval accuracy will be evaluated automatically against ground truth labeled by human on a testing question set containing 30 human labeled questions on 4 testing locations.
- The end-to-end answer accuracy will be evaluated on the same dataset as retrieval accuracy. This evaluation will be conducted by human, giving a score of 5 and a comments pointing out potential issue. The comments are visualized and analyzed through word cloud.

The initial verification process will be shown below:

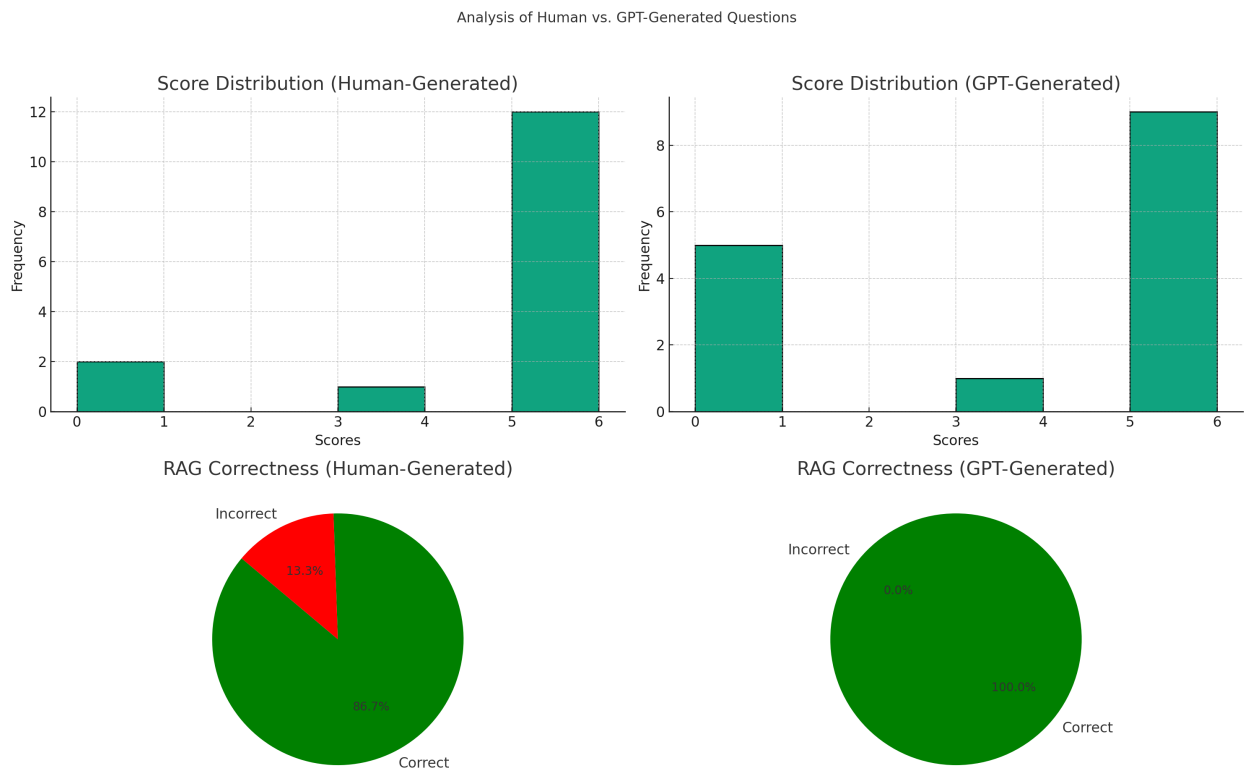


Figure 6: Analysis of End to End Accuracy as well as RAG accuracy

2.4.7 Version Iteration

We'll make a testing dataset to test all aspects of the module, this dataset covers many useful scenarios and will be open-sourced in the following link. In our verification part,

this dataset will play a significant role in evaluating whether this module is working efficiently. We will gradually iterate the version based on the accuracy on the testing dataset. We will also improve the response time step by step.

2.4.8 AI-powered Response Generation Subsystem Interface

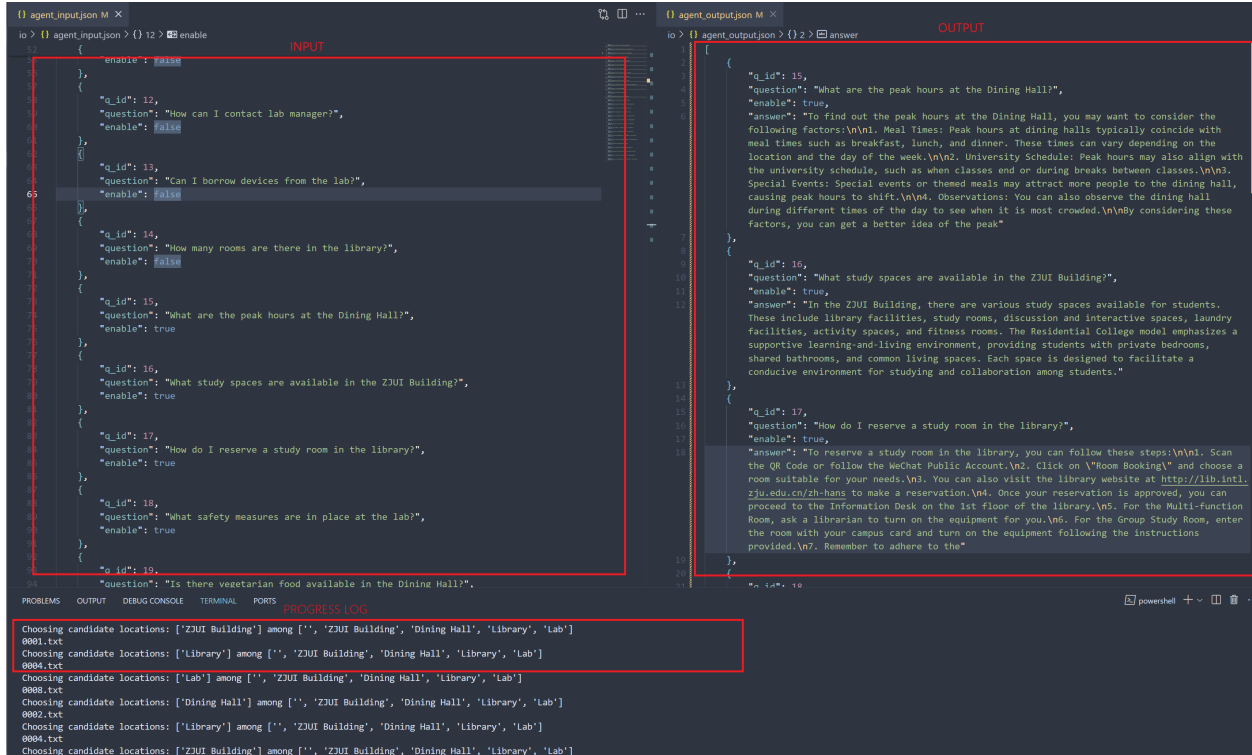


Figure 7: Interface Demo

The code is available at [project_repos](#)

2.5 User Interface

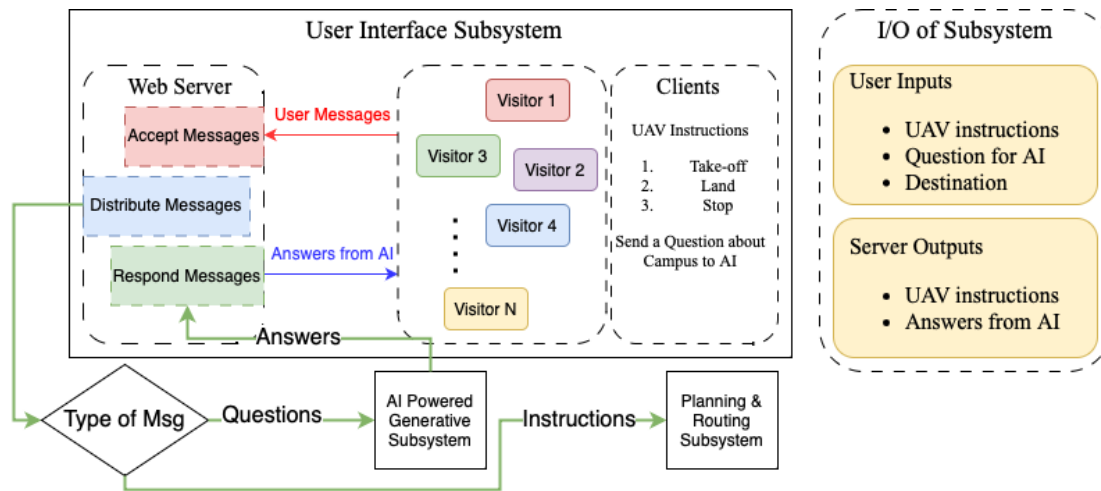


Figure 8: User Interface Diagram

The User Interface (UI) subsystem serves as the primary point of interaction between the users and the AI-powered response generation system. It is designed to be intuitive and user-friendly, enabling users to easily submit queries about the ZJU-UIUC campus and issue commands to the UAV.

2.5.1 Subsystem Architecture and Design

The User Interface (UI) subsystem is the principal conduit for user interactions within the AI-powered response generation system. It is meticulously crafted to be both intuitive and user-friendly, enabling seamless submission of queries and UAV command issuance.

The subsystem is comprised of several key components:

- A web server that accepts and distributes messages.
- Client interfaces for visitors to interact with the system.

The architecture delineates the following operational flow:

1. The web server receives messages from various visitors through the user interface.
2. Depending on the message type, identified as a question or a command, the server routes the message to the respective subsystem for processing.

The design leverages a set of defined APIs to manage the interactions between the UI and other subsystems, promoting real-time processing and ensuring data consistency and reliability. The modular nature of the design allows for scalable and maintainable enhancements, critical for future integration and functionality expansion.

The envisioned deliverables include:



Campus Tour Guide

Destination:

Any questions interested?

Figure 9: Simple UI

- Hosting the web service on a remote server.
- Developing an easy-to-navigate, full-stack framework.
- Establishing a robust connection between the web server and the AI and Planning subsystems.

This architecture is designed to provide a seamless, efficient, and secure user experience, whether it's for informational queries or operational control over the UAV.

2.5.2 Input and Output of the Subsystem

Table 4: Input and Output of the User Interface Subsystem

Field Name	Type	Meaning
UAV Instructions User2Server	Input	Take-off, Land, Stop instruction to UAV
User Questions	Input	Questions about ZJUI Campus
User Destination	Input	User's destination
AI Answer	Output	Answer to user's question
UAV Instructions Server2UAV	Output	Take-off, Land, Stop instruction to UAV

- Finalize hosting web service (Xuanbo, 2024-3-21)
- Build an easy-to-use full-stack framework (Xuanbo and Hao, 2024-3-30)
- Connect the web server with the host server (Xuanbo, 2024-3-21)

The input to the user interface subsystem is **the answer to the user's questions and the status of the drone**. The output of the subsystem is **questions by the user and commands to the drone**.

2.5.3 Frontend Development

The frontend of the UI subsystem is developed using React and JavaScript, offering a dynamic and responsive web interface. The design features a minimalist layout to enhance user experience and facilitate ease of use. Key elements of the UI include:

- **Question Input Block:** A dedicated area where users can type in or voice their queries about the ZJU-UIUC campus.
- **Instruction Buttons:** Several interactive buttons designed to issue predefined commands to the UAV, such as "Take off," "Land," and "Stop". Besides, there is an additional button to send questions to AI-powered Generative System "Send".
- **Campus Image Display:** An image block that dynamically displays photographs of the ZJU campus, which could be relevant to the user's queries or for showcasing UAV functionalities.

This design ensures that users have a straightforward and efficient way to interact with the system, whether seeking information or controlling the UAV.

2.5.4 Remote Server Setup

The subsystem utilizes Ali Cloud for hosting the remote server, establishing a robust and scalable infrastructure. The connection to the server is secured via SSH, ensuring encrypted communication channels. This server plays a critical role in:

- Managing connections between end-users and the Planning & Control subsystem.
- Facilitating data exchange between the UI and AI subsystems, ensuring seamless integration and real-time response capabilities.

2.5.5 Connection with AI Subsystem

The integration between the UI and the AI subsystem is essential for the real-time processing of user queries and drone commands. This connection is established through a well-defined API that:

- Allows for the seamless transmission of user inputs from the UI to the AI subsystem, where they are processed to generate responses or drone commands.
- Enables the AI subsystem to send back the generated responses or status updates directly to the UI, ensuring that users receive immediate and relevant feedback.

This integration is designed to be highly efficient, minimizing latency and maximizing the accuracy and relevance of the information provided to the users.

2.5.6 Verification and Results

The verification of the User Interface Subsystem involved a series of rigorous tests and evaluations to ensure its functionality, usability, and performance met the project requirements. The following methodologies were employed:

1. **Functional Testing:** Comprehensive functional tests were conducted to verify that all features of the UI subsystem, including question input, instruction buttons, and campus image display, worked as intended.
2. **Usability Testing:** Usability testing sessions were conducted with a diverse group of users to evaluate the UI's ease of use, intuitiveness, and effectiveness in facilitating interaction with the AI-powered Campus Tour Guide UAV.
3. **Performance Testing:** Performance tests were carried out to assess the responsiveness and reliability of the UI subsystem under various load conditions, ensuring it could handle multiple concurrent user requests without degradation in performance.
4. **Integration Testing:** Integration tests were performed to validate the seamless communication between the UI subsystem and other project modules, including the AI subsystem and the Planning & Control subsystem.

```
{  
  "userID": "966",  
  "destination": "ZJUI Buidling",  
  "question": "How many staffs are there in ZJUI Building?",  
  "instruction": "Take-off"  
}
```

Figure 10: User Infomation

The results of the verification process are summarized as follows:

1. **Functional Testing:** All features of the UI subsystem were successfully tested and found to be functioning according to specifications. Users were able to input questions, issue instructions to the UAV, and view relevant campus images without encountering any major issues.
2. **Usability Testing:** Feedback from usability testing sessions indicated that users found the UI to be intuitive and easy to navigate. The question input block, instruction buttons, and campus image display were all praised for their clarity and responsiveness.
3. **Performance Testing:** Performance tests demonstrated that the UI subsystem could handle a significant number of concurrent user requests without noticeable slow-downs. Response times remained within acceptable limits even under heavy load conditions.

4. **Integration Testing:** Integration tests confirmed seamless communication between the UI subsystem and other project modules. Data exchange with the AI subsystem for question answering and the Planning & Control subsystem for UAV instructions occurred without errors.

2.6 Planning & Control Subsystem

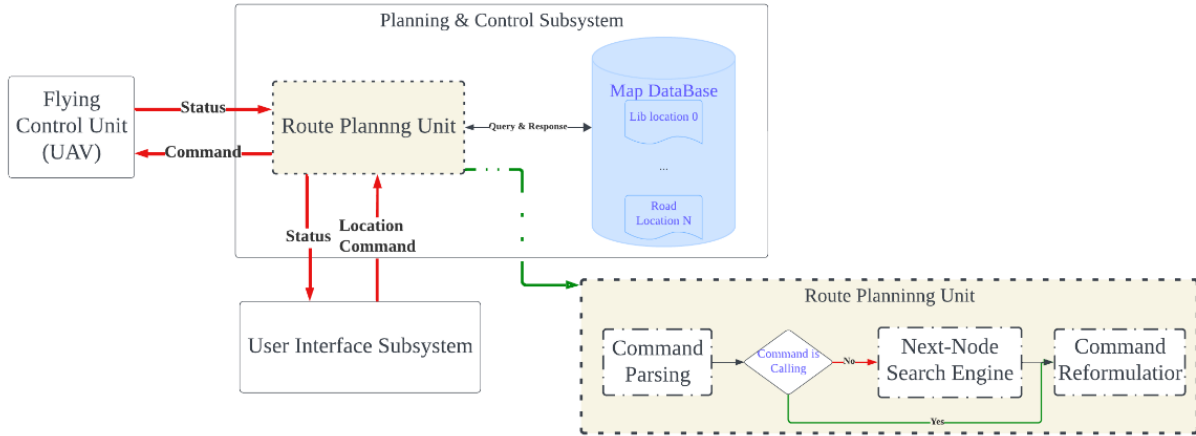


Figure 11: Planning and Control Block Diagram

The Planning and Control Subsystem is an integral component of our UAV operational framework, designed to process commands from the user interface, assess the current status of the drone, and issue precise navigational instructions based on the drone’s specifications. This subsystem interfaces directly with PX4 APIs, an open-source flight control software, to monitor and control the UAV’s flight parameters. The primary input to this subsystem is **the user’s command and the drone’s status**, while its output is the **command to the UAV**, ensuring that each operation is executed safely and efficiently.

2.6.1 Notation and Explanation

Table 5: Notations Used for Planning and Control Subsystem

Name	Meaning
Node	The map is continuous, we extract a sets of locations as nodes
Map Database	A data-store for the node
Next Node	The next node we plan to go to
Link	The minimum route unit linking 2 nodes
Command	An instruction to UAV or from user
Parsed Command	An instruction equivalent to a set of MAVSDK APIs
MAVSDK	a software dev kit consists of APIS instructing UAV
Search Engine	A module searching for next node given current location and command
Reformulation	A process formulating instruction to comply with MAVSDK APIs

2.6.2 Subsystem Architecture and Design

This subsystem obtains a command and it reformulates the command to the drone to execute. The reason why this module is essential and vital is this module helps visitor has a safe and comfortable trip. This subsystem objective is to find a short and comfortable route for user to take while taking the tour inside ZJU-UIUC campus.

This subsystem establishes a robust connection with a remote server to access vital flight data, including starting points, destinations, flight altitudes, and other navigational parameters. This connection is crucial for retrieving real-time information necessary for flight planning and control. The communication between the Planning & Control Subsystem and the remote server is facilitated through a secure, encrypted channel, ensuring the integrity and confidentiality of transmitted data. This setup allows the subsystem to:

- Obtain real-time updates on weather conditions, no-fly zones, and other environmental factors that may affect flight paths.
- Receive user-defined flight parameters such as starting location, destination, and preferred flight height.
- Update the UAV's mission parameters in response to changing conditions or user commands.

2.6.3 Input and Output of the Subsystem

Table 6: Inputs and Outputs

Inputs	Outputs
From the Unmanned Aerial Vehicle (UAV): <ul style="list-style-type: none"> • Latitude and longitude position of the UAV. • Current velocity, acceleration, and attitude (orientation) of the UAV. 	To the Unmanned Aerial Vehicle (UAV): <ul style="list-style-type: none"> • Attitude control commands for the UAV. • Velocity and acceleration control commands for the UAV.
From the AI-Powered Generative System: <ul style="list-style-type: none"> • Responses generated by the AI system based on user queries. 	To the AI-Powered Generative System: <ul style="list-style-type: none"> • User queries directed to the AI system.
From the User Interface System: <ul style="list-style-type: none"> • User commands and requests submitted through the interface. 	To the User Interface System: <ul style="list-style-type: none"> • Responses provided to users based on their queries.

2.6.4 Data Collection and Tagging

The subsystem must use a data store which stores the locations and other related meta-data for each link. A link is a pair of nodes. A link is a viable path. Any connections between nodes that are not linked are not a valid path. For instance, if the connection between 2 nodes cross a lake which visitor is impossible to follow, it won't be our data store.

2.6.5 Algorithms for Subsystem

Let's define the mathematical model for the planning and control subsystem:

- **Nodes N** : A set of extracted locations from the continuous map, which serve as possible waypoints for the UAV.
- **Map Database D** : A datastore that contains the nodes and links information.
- **Links L** : Directed edges between nodes representing the minimum navigable path for the UAV. Each link connects two nodes and has associated costs (like distance, time, or energy consumption).
- **Commands C** : Instructions from the user or system that need to be executed by the UAV.
- **Parsed Commands P** : Translated commands into MAVSDK API calls.
- **MAVSDK M** : The software development kit used to control the UAV.

Given:

- **Current Node n_{current}** : The UAV's current position represented as a node.
- **Destination Node $n_{\text{destination}}$** : The target position the UAV needs to reach.

Objective

Find the optimal path Π from n_{current} to $n_{\text{destination}}$ minimizing the cost function F , which could include factors like distance, time, energy, etc.

Constraints

- The UAV can only travel along links in L from one node to another.
- The path must start at n_{current} and end at $n_{\text{destination}}$.

Dijkstra's Algorithm for Pathfinding

Dijkstra's algorithm can be used to find the shortest path from n_{current} to $n_{\text{destination}}$ in a graph represented by nodes and links.

1. Initialize a priority queue Q with the starting node n_{current} , setting its cost to 0 and all other nodes to infinity.
2. While Q is not empty:

- Extract the node n with the lowest cost from Q .
- If n is $n_{\text{destination}}$, terminate and reconstruct the path.
- For each neighbor n_{next} of n connected by a link l in L :
 - Calculate the tentative cost to reach n_{next} as the sum of the cost to reach n and the cost of l .
 - If the tentative cost is less than the current cost to reach n_{next} , update it and add n_{next} to Q .

The result of Dijkstra’s algorithm is the path Π that optimizes the cost function F , providing an efficient route for the UAV from the starting point to the destination.

Implementation with MAVLink Python Package: The MAVLink Python package provides a comprehensive set of tools for communicating with the UAV, offering functionalities such as:

- Sending navigational commands and mission updates to the UAV.
- Receiving real-time status information, including location, battery level, and flight mode.
- Managing telemetry data to monitor and adjust flight parameters as needed.

This package is instrumental in bridging the gap between high-level operational commands and the low-level directives understood by the UAV, ensuring that the subsystem can effectively translate user intentions into actionable flight paths.

2.6.6 Integration with QGroundControl Platform

QGroundControl (QGC) represents a pivotal tool in the UAV operational toolkit, offering an intuitive and feature-rich ground control station interface for UAV management. This open-source platform provides full flight control and mission planning capabilities for any MAVLink-enabled drone, making it an indispensable asset for our project.

Key Functionalities of QGC: The QGC platform is renowned for its comprehensive suite of functionalities, designed to enhance the operability and efficiency of UAV missions. Some of its most notable features include:

- **Intuitive Flight Planning:** Users can plan missions with ease, specifying waypoints, actions, and objectives directly on a detailed map interface.
- **Real-Time Flight Data Monitoring:** QGC displays live telemetry data, including UAV position, altitude, and battery status, allowing for close monitoring and immediate adjustments.
- **Vehicle Setup Configuration:** The platform provides tools for configuring UAV settings, calibrating sensors, and adjusting flight parameters to optimize performance.

- **Log Analysis:** Post-flight data logs can be analyzed within QGC to assess flight performance, identify potential issues, and refine operational strategies.

Benefits for the UAV System: The integration of QGC into our UAV project brings several significant advantages, enhancing both the development and operational phases:

- **Enhanced Mission Planning:** QGC's user-friendly mission planning tools allow for the precise and efficient design of flight paths, directly contributing to the success of complex UAV operations.
- **Operational Flexibility:** With real-time data monitoring and vehicle configuration capabilities, QGC provides the flexibility needed to adapt to dynamic operational environments and mission requirements.
- **Safety and Compliance:** The platform's comprehensive data logging and analysis features support thorough post-mission reviews, facilitating adherence to safety protocols and regulatory compliance.
- **Streamlined Development:** By leveraging QGC's robust features and MAVLink integration, we can accelerate the development process, focusing on custom functionalities specific to our project's needs without reinventing basic control and monitoring capabilities.

The synergy between QGroundControl and our subsystems significantly enhances our UAV's operational capabilities, enabling sophisticated mission planning, real-time control, and detailed performance analysis. Through this integration, we aim to achieve a level of precision, safety, and efficiency that sets a new standard for UAV operations within the ZJU-UIUC campus context.

2.6.7 Verification and Results

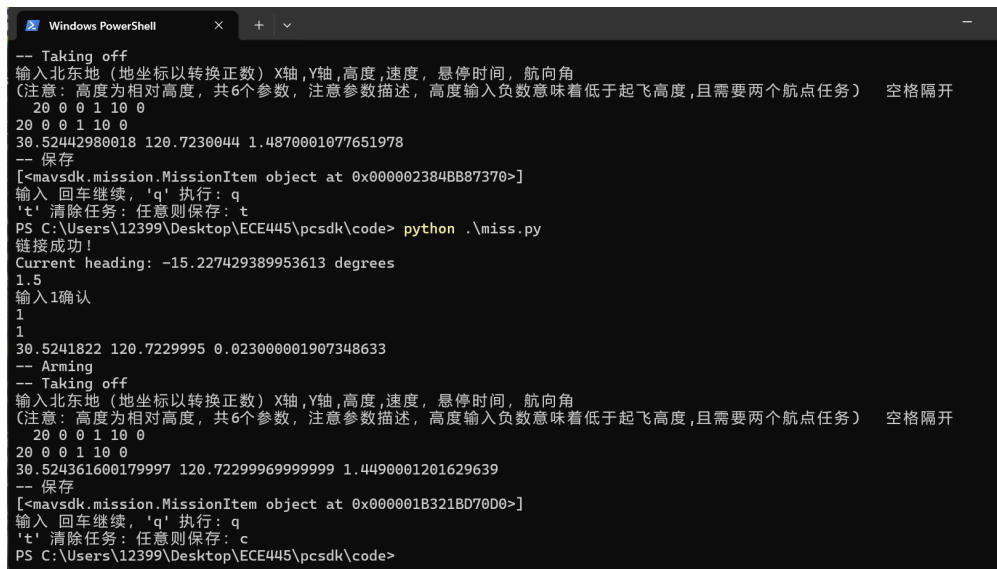
The verification of the Planning & Control Subsystem involved a comprehensive evaluation of its capabilities in interpreting flight missions, generating optimal routes, and controlling the UAV during flight operations. The following methodologies were employed:

1. **Mission Interpretation Testing:** Testing was conducted to ensure that the subsystem could accurately interpret flight missions specified in text files, including waypoints, altitudes, speeds, and pause durations.
2. **Route Generation Testing:** Tests were carried out to verify the efficiency and reliability of the Dijkstra's algorithm implementation in generating optimal flight routes while considering factors such as terrain obstacles and no-fly zones.
3. **Flight Control Testing:** Flight control tests were performed to assess the subsystem's ability to execute flight missions autonomously, including takeoff, landing, and waypoint navigation, using MAVSDK scripts.

4. **Integration Testing:** Integration tests were conducted to validate the subsystem's seamless integration with other project modules, including the User Interface Subsystem, AI subsystem, and Sensing subsystem.

The results of the verification process for the Planning & Control Subsystem are summarized as follows:

1. **Mission Interpretation Testing:** The subsystem successfully interpreted various flight missions specified in text files, accurately translating waypoints and other parameters into executable commands for the UAV.
2. **Route Generation Testing:** Dijkstra's algorithm consistently generated optimal flight routes, avoiding obstacles and adhering to operational constraints. The subsystem demonstrated reliability in finding efficient paths even in complex environments.
3. **Flight Control Testing:** Flight control tests confirmed the subsystem's capability to execute flight missions autonomously, including takeoff, landing, and waypoint navigation. MAVSDK scripts facilitated smooth operation and precise control of the UAV.
4. **Integration Testing:** Integration tests revealed seamless communication between the Planning & Control Subsystem and other project modules. The subsystem effectively received user-defined destinations from the UI subsystem, integrated AI-generated insights, and utilized real-time environmental data from the Sensing subsystem for adaptive flight planning.



```
Windows PowerShell
-- Taking off
输入北东地 (地坐标以转换正数) X轴, Y轴, 高度, 速度, 悬停时间, 航向角
(注意: 高度为相对高度, 共6个参数, 注意参数描述, 高度输入负数意味着低于起飞高度, 且需要两个航点任务) 空格隔开
20 0 0 1 10 0
30.52442980018 120.7230044 1.4870001077651978
-- 保存
[<mavsdk.mission.MissionItem object at 0x00002384BB87370>]
输入 回车继续, 'q' 执行: q
't' 清除任务: 任意则保存: t
PS C:\Users\12399\Desktop\ECE445\pcsdk\code> python .\miss.py
链接成功!
Current heading: -15.227429389953613 degrees
1.5
输入1确认
1
1
30.5241822 120.7229995 0.02300001907348633
-- Arming
-- Taking off
输入北东地 (地坐标以转换正数) X轴, Y轴, 高度, 速度, 悬停时间, 航向角
(注意: 高度为相对高度, 共6个参数, 注意参数描述, 高度输入负数意味着低于起飞高度, 且需要两个航点任务) 空格隔开
20 0 0 1 10 0
20 0 0 1 10 0
30.524361600179997 120.72299969999999 1.4490001201629639
-- 保存
[<mavsdk.mission.MissionItem object at 0x00001B321BD70D0>]
输入 回车继续, 'q' 执行: q
't' 清除任务: 任意则保存: c
PS C:\Users\12399\Desktop\ECE445\pcsdk\code>
```

Figure 12: UAV Flight mission by Planning & Routing Subsystem

2.7 Sensor System

This section provides an overview of the sensor subsystem, detailing the specific sensors used for measuring humidity, temperature, and angular rate. Each sensor's type, model, and primary functionality are outlined to give a clear understanding of how they contribute to the overall system.

2.7.1 Introduction

We want to connect a temperature and humidity sensor to a PCB board, powered by a drone, with the sensor's temperature and humidity measurements displayed on an external LCD screen. This way, as the drone follows the user, the user can view real-time weather conditions.

Here is the schematic diagram of our overall sensor section. From the diagram, we can see that the entire circuit is controlled by the STM chip. Firstly, it converts the 5V regulated power supply from the aircraft input to a 3.3V voltage. Then, it connects to the temperature and humidity sensors as well as the speed sensor to obtain data. Subsequently, the acquired data is transmitted to the LCD display screen by the microcontroller for displaying the results.

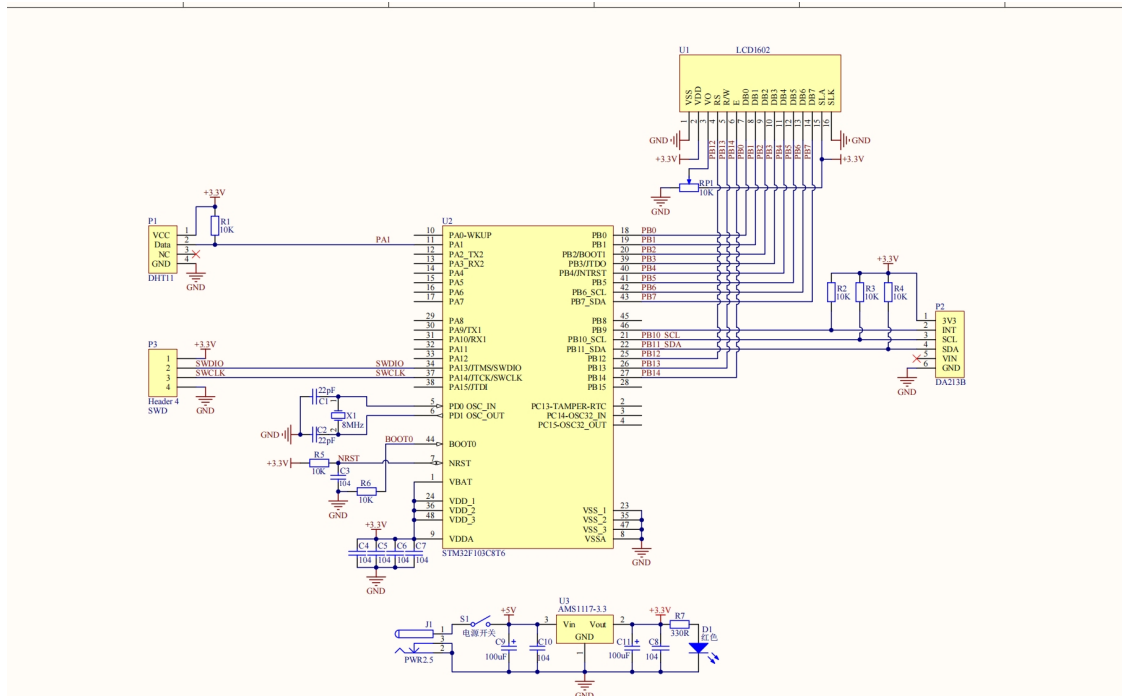


Figure 13: Circuit Design

2.7.2 Temperature and Humidity Sensor

- Type & Model: DHT-11

- **Functionality:** We selected the DHT11 sensor mainly because of its simplicity, ease of use, and low cost, as well as its ability to accurately measure environmental temperature and humidity. Its digital output makes data reading and processing more convenient, and it offers high accuracy and stability, providing reliable temperature and humidity measurements in various environmental conditions to ensure flight safety and stability for our project.[4]

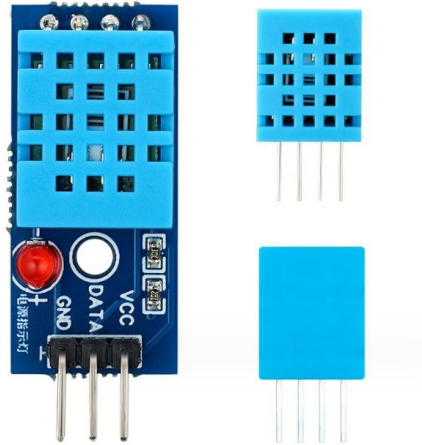


Figure 14: DHT 11 Outside view

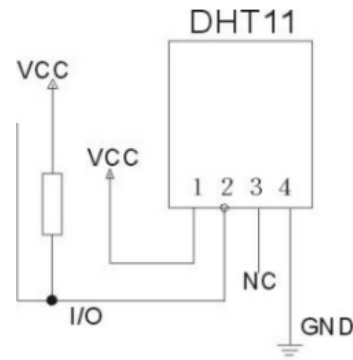


Figure 15: DHT11 pin diagram

2.7.3 Accelerometer Sensor

- **Type & Model:** DA213B

- **Functionality:** The DA213B accelerometer sensor was chosen primarily for its high precision, sensitivity, and stability. Capable of accurately measuring acceleration in three axial directions with digital output, it offers convenient data reading and processing. Additionally, its low power consumption and small footprint make it suitable for embedded systems and mobile device applications. In our project, the DA213B sensor provides precise acceleration data for attitude control and motion tracking, enhancing flight stability and accuracy.[5]

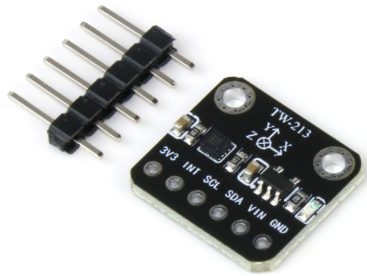


Figure 16: DA213B Outside view

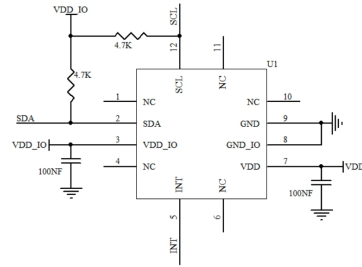
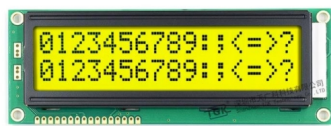


Figure 17: DA213B pin diagram

2.7.4 LCD Display

- **Type & Model:** LCD1602

- **Functionality:** The LCD1602 screen was selected primarily for its simplicity and affordability. It can display 16 columns and 2 rows of characters, providing basic text display functionality. Utilizing the standard HD44780 controller, it communicates easily with microcontrollers through a simple interface. Moreover, the LCD1602 screen has low power consumption, making it suitable for embedded systems and mobile devices. In our project, it provides a convenient way to display sensor data, system status, and user interface elements, simplifying user interaction and making system operation more intuitive.



TM162 G-1 (16X2)
 SIZE:122.0X44.0mm
 VA:99.0X44.0mm

Figure 18: LCD1602 Outside view

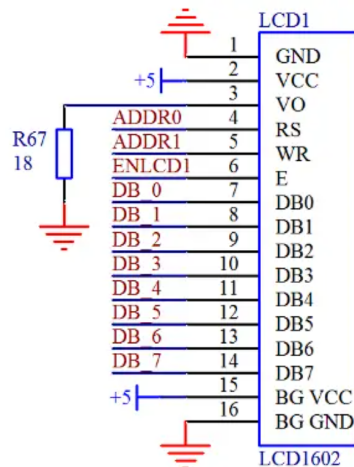


Figure 19: LCD1602 pin diagram

2.7.5 PCB design

Here is our PCB design diagram. We have designed the PCB board to be 40cm * 60cm in size to match the dimensions of the drone. Two sensors are directly soldered onto the PCB board, while the LCD screen is connected externally using ribbon cables. The input terminal of the PCB board is supplied with 5V voltage from the drone, and the output terminal is connected to the LCD screen.

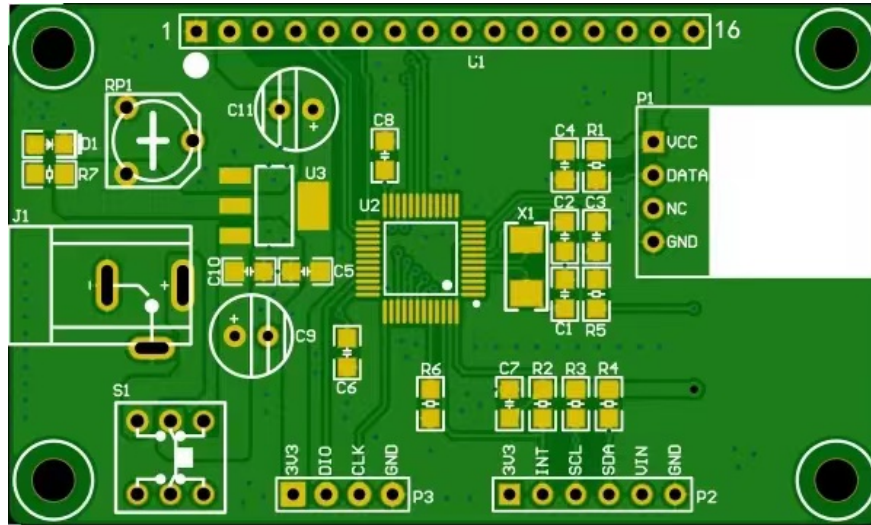


Figure 20: PCB Design

2.7.6 Verification and Requirement Plan

There are several functionalities we want to achieve for the sensor unit:

1. The stability of converting 1.5V voltage to 3V voltage.
2. Whether the two sensors can operate properly, achieve data measurement, and transmit data.
3. Whether the LCD screen can display normally and show the data we need in real-time.

We plan to verify these goals using the following verification methods.

1. Utilize a stable power supply to provide a 1.5V input voltage, connecting it to the voltage conversion circuit. Measure the output voltage using an oscilloscope or digital multimeter and record its value. Continuously observe the stability of the output voltage over a period of time (e.g., 30 minutes to 1 hour), noting any fluctuation range. Repeat the test steps to check the stability of the circuit under different temperature and load conditions.
2. Connect the sensors to the test board and use an oscilloscope or microcontroller to read the sensor's output data. Simulate sensor operation under different environmental conditions, such as varying temperature, humidity, and light levels, and

observe the sensor's response. Verify whether the sensor output data matches the expected values and record any anomalies. Test the data transmission functionality to ensure that sensor data can be successfully transmitted to the microcontroller or other devices and correctly interpreted.

3. Connect the LCD screen to the test board and provide appropriate power. Send test data to the LCD screen and observe whether the screen displays correctly. Check whether the LCD screen can display various types of data, including text, numbers, and graphics. Under simulated working conditions such as temperature changes and vibration, verify whether the LCD screen can stably display data. If possible, conduct long-term testing to confirm the stability and durability of the LCD screen.

2.8 Tolerance Analysis

Ensuring the robustness of the UAV system involves conducting a thorough tolerance analysis. This analysis focuses on identifying potential vulnerabilities within the system, assessing risks, and devising strategies to mitigate these risks. Two critical areas of concern include route planning stability and communication system delays.

2.8.1 Route Planning Stability

Route planning is susceptible to various environmental factors that can affect the UAV's ability to navigate effectively. These factors include wind, physical obstacles, and the presence of visitors within the campus.

Risks:

- Wind can significantly alter the UAV's course, leading to deviations from the planned route and potentially unsafe conditions.
- Obstacles such as trees and buildings may not only hinder the UAV's path but also pose a risk of collision.
- Visitors moving unpredictably through the campus can introduce dynamic variables, complicating the UAV's navigation and safety protocols.

Wind's impact on UAV navigation can be analyzed using vector mathematics, specifically the wind triangle theory. The ground speed vector (\vec{V}_g) of the UAV is the vector sum of its airspeed vector (\vec{V}_a), which is the speed and direction relative to the air, and the wind speed vector (\vec{V}_w), which represents the speed and direction of the wind. This relationship is given by:

$$\vec{V}_g = \vec{V}_a + \vec{V}_w \quad (1)$$

2.8.2 Communication System Delay

Delays in the communication system can arise in two main interactions: 1) between the user and the remote server, and 2) between the remote server and the planning & control subsystem.

Risks:

- Delays in user commands reaching the UAV could result in outdated or inappropriate actions being taken, especially in fast-changing environments.
- Latency in the planning & control subsystem receiving data from the remote server may lead to decision-making based on stale information, compromising operational safety and efficiency.

Communication delays in UAV operations can be quantitatively analyzed using the concept of network latency, which encompasses the time it takes for a signal to travel from the

source to the destination and back. This delay, represented by Δt , can significantly affect real-time decision-making and UAV control, particularly in dynamic environments.

The total communication delay (T_{total}) experienced in the system can be modeled as the sum of various components, including user interface to remote server latency ($T_{\text{user-server}}$), server processing time ($T_{\text{server-proc}}$), and remote server to UAV latency ($T_{\text{server-UAV}}$):

$$T_{\text{total}} = T_{\text{user-server}} + T_{\text{server-proc}} + T_{\text{server-UAV}} \quad (2)$$

Each component of this model can be independently measured and optimized to reduce the overall delay in the communication system.

The operational risks posed by communication delays can be further examined through the lens of control system theory, particularly in terms of the impact on the UAV's control loop stability. A delayed control signal can be modeled in the time domain as:

$$u(t) = K \cdot e(t - \Delta t) \quad (3)$$

where $u(t)$ is the control signal at time t , K is the gain, $K = 1$ if the signal is received and $K = 0$ instead, $e(t - \Delta t)$ represents the error observed at a delayed time, and Δt is the delay.

2.8.3 GPS Locating Error

Global Positioning System (GPS) technology is crucial for UAV navigation, offering real-time location data that guides the UAV's flight path. However, GPS signals can be subject to interference from environmental factors, such as atmospheric conditions, buildings, and signal jamming, leading to potential errors in location accuracy.

Risks:

- **Mission Failure:** Critical missions requiring precise location data, such as aerial photography or targeted delivery, could fail due to inaccurate positioning.

GPS locating error (E_{gps}) can be influenced by several factors, including signal propagation delay, atmospheric conditions, and multipath errors. The total error can be modeled as a combination of these factors:

$$E_{\text{gps}} = E_{\text{propagation}} + E_{\text{atmospheric}} + E_{\text{receiver}} \quad (4)$$

Where:

- $E_{\text{propagation}}$ represents the error due to signal propagation delay.
- $E_{\text{atmospheric}}$ accounts for delays caused by atmospheric conditions (ionospheric and tropospheric delays).
- E_{receiver} represents the error due to inaccuracies in the GPS receiver itself.

This model highlights the multifaceted nature of GPS errors and underscores the importance of comprehensive testing and calibration to minimize overall error in UAV navigation systems.

3 Cost and schedule

3.1 Cost Analysis

3.1.1 Labor

The labor cost is calculated based on the working hours and wage pricing of each team member. We set the hourly wage at 100 RMB based on market research and the skill levels of team members. Considering the total project duration of 8 weeks with 40 hours of work per week, the total working hours per team member are:

$$40 \text{ hours/week} \times 8 \text{ weeks} = 320 \text{ hours}$$

Therefore, the labor cost per team member is:

$$100 \text{ RMB/hour} \times 320 \text{ hours} = 32000 \text{ RMB}$$

We chose an hourly wage of 100 RMB, which is based on market wage levels and the skill and experience levels of team members. According to survey data from the Institute of Electrical and Electronics Engineers (IEEE) [6], the average salary for graduates in Electrical and Computer Engineering (ECE) is around 200,000 RMB per year. Calculated on a full-time basis, the average hourly wage is approximately 100 RMB. Thus, our pricing aligns with market rates and incentivizes team members to dedicate sufficient time and effort while maintaining cost-effectiveness.

3.1.2 Parts

If the duration of the project is 8 weeks, we need to consider the costs of ChatGPT4 API and Simple Application Server, which are billed monthly, for 2 months.

ChatGPT4 API has a monthly cost of 240 RMB, multiplied by 2 months equals 480 RMB.

Simple Application Server has a monthly cost of 49 RMB, multiplied by 2 months equals 98 RMB.

The table below provides a breakdown of the parts and their estimated costs:

Now let's recalculate the total cost:

$$\begin{aligned} &5174 + 680 + 10 \times 2 + 15 \times 2 + 20 + 50 \\ &+ 480 + 98 + 18 \times 6 = 6819 \text{ RMB} \end{aligned}$$

Therefore, the total cost is 6819 RMB.

Description	Manufacturer	Part #	Quantity	Cost (RMB)
Drone	PixHawk	MFP450	1	5174
Mavlink Module	Amovlab	-	1	680
Temperature Sensor	Aosong	DHT11	2	10
Acceleration Sensor	MiraMEMS	DA213B	2	15
LCD Screen	Touglesy	LCD1602	1	20
PCB Board	Custom	-	1	50
Simple Application Server	Alibaba Cloud	-	1	49 per month
ChatGPT4 API	OpenAI	-	1	240 per month
Battery	Nanfu	-	6	18

Table 7: Parts List and Estimated Costs

3.1.3 Grand Total

The grand total cost of the project can be calculated by summing up the labor cost and the cost of parts:

- Labor: 32,000 RMB
- Parts: 6,819 RMB

Grand Total: $32000 + 6819 = 38,819$ RMB

3.2 Schedule & Milestone

3.2.1 Milestone 1: Basic Hardware Configuration and Functionality

This milestone focuses on establishing the foundational hardware capabilities of the UAV, ensuring it can perform basic functions independently.

Drone Control and Movement: The first objective is to achieve independent movement and control of the drone. This involves configuring the propulsion system, including motors and propellers, and integrating them with the flight controller. Key considerations include: Calibration of electronic speed controllers (ESCs) to ensure responsive and stable flight. Implementation of a basic manual control system, possibly using a radio transmitter and receiver, to test movement capabilities.

GPS Integration: The integration of the GPS module aims to enable the UAV to sense its location accurately. This functionality is critical for navigation and will later support advanced features such as autonomous flight. Aspects to detail could include: Selection of

a GPS module that offers high accuracy and fast lock-on times. Strategies for minimizing GPS signal interference, ensuring reliable location data.

Sensor Powering and Testing: Ensuring that all onboard sensors can be powered and transmit data correctly is crucial at this stage. This involves: Establishing a power distribution system that meets the energy requirements of each sensor without affecting the UAV's overall power balance. Conducting initial tests to verify that sensors (e.g., depth, temperature & humidity, air pressure) are operational and accurately capturing environmental data.

3.2.2 Milestone 2: Software Integration and Functional Testing

Milestone 2 advances the UAV project by integrating software controls with the previously established hardware framework, allowing for more sophisticated operations.

Control Subsystem Integration: The key objective is enabling the drone to be controlled via the software-based control subsystem. This involves: Developing or customizing software that can send commands to the flight controller, adjusting parameters such as altitude, speed, and direction based on real-time inputs. Ensuring the control subsystem can process inputs from the user interface and translate them into actionable commands for the drone.

Data Reception from GPS and Sensors: At this stage, it's crucial that the control subsystem can receive and process data from the GPS module and other sensors. Considerations include: Implementing communication protocols that allow for the efficient transmission of sensor data to the control subsystem. Integrating GPS data to enhance flight planning capabilities, allowing the UAV to adjust its route based on location information.

Route Planning and Simulation: The ability to output routes based on sensor and GPS data is a critical step towards autonomous flight. While actual drone movement isn't required at this milestone, the focus should be on: Developing algorithms that can calculate optimal flight paths based on input data (e.g., avoiding obstacles, minimizing energy consumption). Simulating these routes in a controlled environment to validate the planning logic and identify potential improvements.

3.2.3 Milestone 3: Basic Autonomy and Obstacle Awareness

Milestone 3 aims to enhance the UAV's autonomy by enabling it to follow a human subject and navigate from point A to point B without obstacles, incorporating initial capabilities to recognize and plan routes around obstacles.

Human Following Without Obstacles: The system will be developed to recognize and follow a human subject using image processing or motion detection technologies. This feature requires: Integration of camera sensors with real-time video processing software to identify and track the human subject. Algorithms for dynamic adjustment of the UAV's flight path to maintain a constant distance to the subject.

Point-to-Point Flight with Dynamic Speed Adjustment: The UAV will autonomously navigate between designated points A and B, implementing logic to slow down or stop if the human subject moves beyond a predefined distance. This involves: Developing waypoint navigation software that utilizes GPS coordinates for precise location tracking. Implementing distance monitoring algorithms to dynamically adjust the UAV's speed based on the subject's proximity.

Obstacle Recognition and Avoidance Planning: Introducing the capability to detect obstacles and compute alternative routes. Key development steps include: Utilizing depth sensors and image recognition to identify obstacles within the UAV's flight path. Integrating path planning algorithms (e.g., A* or Dijkstra) to calculate detours around obstacles.

3.2.4 Milestone 4: Advanced Autonomy with Obstacle Navigation

Milestone 4 builds on the previous capabilities, enabling the UAV to navigate complex environments with obstacles, and introduces predefined route selection.

Obstacle Navigation and Human Proximity Response: Enhancing the UAV's software to handle dynamic obstacles while maintaining awareness of the human subject's location. This requires: Advanced obstacle detection and avoidance algorithms capable of real-time adjustments to the flight plan. Refined human tracking algorithms to ensure the UAV can safely navigate around obstacles without losing the subject.

Predefined Route Selection: The system will offer a selection of starting and ending points for navigation, improving the UAV's versatility for different mission scenarios. Implementation includes: A database of GPS coordinates representing various predefined routes. A user interface allowing selection from available starting and ending point pairs.

3.2.5 Milestone 5 [Optional]: Interactive Communication and Control

The optional Milestone 5 focuses on enhancing user interaction with the UAV through a web application, voice commands, and interactive chatting.

Web Application for Signal Transmission: Developing a simple web application that enables users to send commands to the UAV. This involves: Creating a user-friendly inter-

face for command input. Establishing a secure and reliable communication link between the web app and the UAV's control system.

Voice Command Reception and Destination Setting: Integrating voice recognition technology to allow users to issue commands and set destinations vocally. Key components include: Voice recognition software capable of processing and interpreting user commands. Software logic to translate vocal commands into navigational actions or destination coordinates.

Interactive Chatting for Environmental Awareness: Implementing a chat interface that enables the UAV to communicate with the user about its surroundings, enhancing situational awareness. This feature requires: Integration of natural language processing (NLP) algorithms to understand and respond to user queries. Access to real-time sensor data, allowing the UAV to provide updates on environmental conditions or navigational status.

3.3 Weekly schedule

Week	Hao Ren's Tasks	Xuanbo Jin's Tasks
Week 1	Research and plan software architecture. Define requirements for drone control system.	Research and plan user interface design. Define requirements for software integration with hardware.
Week 2	Develop software architecture for drone control system. Begin implementation of basic control functionalities.	Design user interface for system control and monitoring.
Week 3	Continue development of drone control software. Implement communication protocols for receiving data from sensors and GPS module.	Implement functionalities for real-time visualization of sensor data. Develop algorithms for data processing and analysis.
Week 4	Test and debug drone control software. Integrate sensor and GPS data with control system.	Test and debug user interface functionalities. Integrate software components for seamless interaction with hardware.
Week 5	Refine software algorithms for improved drone control and navigation. Perform system integration testing.	Refine user interface design based on feedback. Conduct user acceptance testing for software features.
Week 6	Conduct thorough testing and optimization of software. Address any remaining bugs or issues.	Conduct final testing of user interface functionalities. Prepare documentation for software usage and troubleshooting.
Week 7	Finalize software development. Prepare for final demo.	Finalize user interface design. Prepare for final demo.
Week 8	Document final software implementation and configurations. Prepare project documentation for submission.	Document final user interface design and functionalities. Prepare project documentation for submission.

Table 8: Weekly Schedule (Software)

Week	Yuntong Gu's Tasks	Weiang Wang's Tasks
Week 1	Research and plan hardware components required for drone control. Procure necessary hardware components.	Research and plan sensor integration with control system. Procure necessary sensor components.
Week 2	Configure hardware components for drone assembly. Test individual hardware components for functionality.	Set up sensors for testing. Develop initial testing procedures.
Week 3	Set up and calibrate drone control system hardware. Ensure proper communication between components.	Conduct initial tests on sensors to verify functionality. Begin data collection and analysis.
Week 4	Conduct movement tests on assembled drone. Evaluate control system performance. Address any hardware issues.	Conduct calibration tests on sensors. Optimize sensor readings and data accuracy.
Week 5	Integrate hardware components for system testing. Conduct integration tests with software control system.	Integrate sensor data with control system. Test data communication and synchronization.
Week 6	Troubleshoot and debug hardware issues. Conduct stress tests on hardware components. Optimize system performance.	Analyze sensor data for consistency and accuracy. Fine-tune data processing algorithms.
Week 7	Finalize hardware setup and configurations. Prepare for final demonstration and presentation.	Finalize sensor integration with control system. Prepare for final demonstration and presentation.
Week 8	Document final hardware configurations and setup. Prepare project documentation for submission.	Document final sensor integration process and configurations. Prepare project documentation for submission.

Table 9: Weekly Schedule (Hardware)

4 Ethics and Safty

4.1 Ethical Considerations

The development and deployment of the AI-guided tour guide drone raise important ethical considerations that must be addressed. One key concern is privacy. The use of drones for guided tours may intrude upon individuals' privacy rights, particularly if they capture images or videos without consent. It's imperative to prioritize privacy and ensure that data collection and usage adhere to ethical standards and legal regulations.[7]

Fairness and non-discrimination are also crucial ethical considerations. AI algorithms utilized in the tour guide system must be designed and trained to avoid bias and ensure equitable treatment for all users, irrespective of their characteristics. This aligns with the principle of upholding human dignity and equality.

Additionally, there's a responsibility to ensure the safety and well-being of participants during guided tours. The drone will be equipped with advanced obstacle detection and avoidance systems to minimize collision risks and prioritize user safety. Battery safety protocols will be strictly enforced to prevent accidents, including regular inspection, proper storage, and careful handling of lithium polymer (LiPo) batteries.

Furthermore, it's essential to minimize the environmental impact of drone operations. The project will adhere to sustainability principles and employ eco-friendly practices wherever possible to reduce environmental harm and fulfill ethical obligations towards environmental stewardship.

4.2 Safety Considerations

Safety is paramount in the development and operation of the AI-guided tour guide drone. Several safety measures will be implemented to mitigate risks and ensure the well-being of users and developers.

Firstly, the drone's hardware and software systems will undergo rigorous testing and validation to ensure their reliability and stability. Emergency shutdown protocols will be in place to address malfunctions or emergencies promptly.

Secondly, strict battery safety protocols will be enforced to prevent accidents related to lithium polymer (LiPo) batteries. This includes regular inspection, proper storage, and careful handling to minimize the risk of fire or explosion.

A comprehensive Safety Manual will be developed to outline procedures for safe operation and maintenance of the project, covering pre-flight checks, emergency response protocols, battery handling guidelines, human-drone interaction protocols, and maintenance procedures. All project personnel will be required to familiarize themselves with the Safety Manual and adhere to its protocols to ensure a safe working environment.

References

- [1] OpenAI, J. Achiam, S. Adler, *et al.*, *Gpt-4 technical report*, 2024. arXiv: 2303.08774 [cs.CL].
- [2] L. Weng, "Prompt engineering," *lilianweng.github.io*, Mar. 2023. [Online]. Available: <https://lilianweng.github.io/posts/2023-03-15-prompt-engineering/>.
- [3] P. Zhao, H. Zhang, Q. Yu, *et al.*, *Retrieval-augmented generation for ai-generated content: A survey*, 2024. arXiv: 2402.19473 [cs.CV].
- [4] Aosong Electronics Co., Ltd. "Aosong Product 21." (2024), [Online]. Available: <http://www.aosong.com/products-21.html> (visited on 04/19/2024).
- [5] MiraMEMS. "MiraMEMS Product 1." (2024), [Online]. Available: <http://www.miramems.com/product-1.html> (visited on 04/19/2024).
- [6] "IEEE (Institute of Electrical and Electronics Engineers) Salary Survey," 2022.
- [7] IEEE. "'IEEE Code of Ethics'." (2016), [Online]. Available: <https://www.ieee.org/about/corporate/governance/p7-8.html> (visited on 02/08/2020).

Appendix A Requirement & Verification Table

Table 10: Subsystem Index Table

Subsystem Name	Subsystem Index(S-Index)
AI Powered Response Generation Subsystem	1
User Interface Subsystem	2
Planning and Control Subsystem	3
Sensor Unit Subsystem	4

Table 11: Requirement & Verification Table

S-Index	Requirement	Verification	Points
1	Detect user intent with at least 25% accuracy	Test with a diverse set of input queries and verify the accuracy of intent detection.	2
1	Detect user intent with at least 50% accuracy	Continue testing and refining to achieve higher accuracy.	2
1	Generate response within 60 seconds	Measure retrieval time with various queries to ensure performance within the initial time limit.	2
1	Generate response within 40 seconds	Measure retrieval time with various queries to ensure performance within the initial time limit.	2
1	Generate response within 30 seconds	Optimize system to improve performance and meet the final time requirement.	1
1	Can fetch correct external material with 20% accuracy	Testing the agent's RAG accuracy by using manually labeled dataset containing 30 questions for 4 testing locations.	1
1	Can fetch correct external material with 50% accuracy	Testing the agent's RAG accuracy by using manually labeled dataset containing 30 questions for 4 testing locations.	2
1	Can fetch correct external material with 70% accuracy	Testing the agent's RAG accuracy by using manually labeled dataset containing 30 questions for 4 testing locations.	1

Continued on next page

Table 11 continued from previous page

S-Index	Requirement	Verification	Points
1	Generated answers must match the user's intent with an accuracy of at least 25% in given dataset	Compare generated answers from the dataset containing 30 questions with human labeled answer to calculate initial accuracy.	3
1	Generated answers must match the user's intent with an accuracy of at least 50% in given dataset	Compare generated answers from the dataset containing 30 questions with human labeled answer to calculate initial accuracy.	3
2	The web server must handle and route messages with less than 2 seconds latency	Test message routing on the server under load and measure latency	2
2	The client interface must provide intuitive access for users to submit queries and control the UAV	Conduct usability testing with participants to assess ease of use and intuitiveness	1
2	UAV command buttons must send correct instructions to the UAV subsystem with 100% accuracy	Test each button and verify that the correct command is sent to the UAV subsystem	1
2	The web server must send instructions and questions separately to different hosts	Test two hosts if they receive correct messages	1
2	Obtain the user's GPS position as the starting position	Test if the two hosts receive the GPS signal	1
2	The host which process the questions can display answers correctly	Check if the UI can display reasonable answers	2
2	The host which process the instructions can send commands to UAV	UAV can take-off, Stop, Continue, Land correctly and in time	2
3	Must accurately process user commands and drone status within 10 second	Perform stress testing with simultaneous user commands and verify response time	3
3	Must accurately process user commands and drone status within 1 second	Perform stress testing with simultaneous user commands and verify response time	3
3	Must optimize the UAV route based on the current status	Test with different scenarios (no-fly zones, different areas) to verify route optimization	3

Continued on next page

Table 11 continued from previous page

S-Index	Requirement	Verification	Points
3	Should maintain a secure and encrypted connection to the remote server	Verify the encryption standards and conduct penetration testing to assess security	1
3	Must integrate seamlessly with the PX4 APIs for flight control	Execute a series of flight tests to ensure proper integration and control	1
4	Power supply successfully power the hardware unit	Power supply LED works correctly	1
4	Sensors can operate properly	Connect the sensors to the test board Use oscilloscope to read the output data.	2
4	LCD screen can display normally	Connect the LCD screen to the test board Provide appropriate power. Send test data to the LCD screen Check if the screen displays correctly.	2
1,2 3,4	End to End works correctly	Can complete a guide for appointed locations while interacting with visitors	5