Autonomous Drones Based on LiDAR for Environmental Monitoring

Project Report

Submitted by: Your Name

Institution Name

Date: June 14, 2025

Dedication

To those who inspire innovation and environmental stewardship.

Acknowledgment

We express our gratitude to all who supported this project, including our advisors, team members, and the open-source community.

Abstract

This report details the development of a low-cost, customizable quadrotor drone utilizing LiDAR for autonomous navigation and environmental monitoring. The project focuses on creating an affordable platform for public space surveillance, leveraging ROS, Gazebo, and a Pixhawk 2.4.8 flight controller. Key components include hardware design, software integration, and simulation-based validation, with future goals of incorporating AI for waste detection.

Contents

Figure Table

List of Figures

Table List

List of Tables

General Introduction

Ensuring the cleanliness and safety of public spaces such as parks, campuses, and urban streets poses an ongoing challenge for municipalities and institutions. Traditional human patrols, while diligent, are labor-intensive, costly, and often fall short in covering expansive or hard-to-reach areas efficiently. To address this, autonomous aerial systems have emerged as a viable and scalable alternative.

This project introduces the design and implementation of an autonomous surveillance drone system, aimed at detecting and documenting violations such as littering or improper waste disposal. Beyond mere detection, the drone is engineered to capture visual evidence, including the perpetrator's face, enabling automated identification and enforcement of environmental regulations through penalty issuance. This approach replicates the workflow of a human environmental inspector, offering a comprehensive surveillance solution.

This report details the entire process of designing and developing the autonomous drone, with artificial intelligence and computer vision components delegated to a separate team. The document is structured into three chapters:

- Project Scope and Overview of Autonomous Drones Based on Advanced Sensing, outlining the objectives and technological foundation.
- Mechanical and Embedded System Design, covering the hardware and software architecture.
- Autonomous Navigation Simulation, exploring virtual testing and performance validation.

Chapter 1

Project Scope and Overview of Autonomous Drones Based on LiDAR

1.1 Introduction

The development of autonomous drones has been steadily taking shape, marking a significant leap in aerial technology designed to operate without constant human input. These systems are engineered with advanced self-guiding capabilities, allowing them to navigate, adapt, and perform tasks with minimal oversight. This project focuses on crafting such a drone, tailored for surveillance to tackle challenges like maintaining clean public spaces, with features that evolve through testing and refinement (**?**).

What sets this drone apart is its flexibility, making it a pretty versatile tool across a range of fields. From monitoring vast parks and campuses to assisting in disaster response or agricultural oversight, its ability to adjust to different environments and missions is a game-changer. This chapter lays the groundwork by exploring the scope of the project, the drive behind its autonomous design, and the broad applications that highlight its potential.

1.2 Host Company

1.3 Review of Key Technologies for LiDAR-Based Autonomous Drones

LiDAR technology stands out remarkably as a foundational element for autonomous navigation, empowering drones to map their surroundings with precision and navigate confidently. Its ability to detect obstacles and construct detailed environmental models makes it indispensable for ensuring the safety and cleanliness of public spaces. This section explores the operational principles of LiDAR, its specialized applications indoors, and its adaptability for both indoor and outdoor use, drawing on established research to guide our project's direction.

1.3.1 Principles of LiDAR Operation

LiDAR, or Light Detection and Ranging, stands as a marvel of modern sensing technology, operating by emitting laser pulses that reflect off surrounding objects and return to the system. The time it takes for these pulses to bounce back is meticulously measured, enabling the creation of a detailed 3D point cloud that maps the environment with striking accuracy. This process relies on a sophisticated interplay of a light source, a scanning mechanism, and a receiver, working together with a keen eye to capture spatial data. For our drone, this real-time mapping capability is a game-changer, allowing it to deftly navigate around obstacles like trees or indoor furniture without mishap. Its speed and precision make it an ideal backbone for autonomous systems, ensuring safe and efficient movement across diverse terrains (1).

1.3.2 LiDAR Technologies for Indoor Environments

Indoor LiDAR tech is tailored for tight spaces like labs or hallways, where our drone needs to navigate around furniture or walls without bumping into anything. These systems often use shorter-wavelength lasers—around 850 nm or 905 nm—because they're great for short-range accuracy, usually up to 20–30 meters. They create detailed 2D maps by sweeping a laser in a single plane, which is enough for dodging obstacles in a room. They're small, light, and don't need much power, so they're ideal for a lightweight quadrotor like ours, especially for indoor testing in a controlled lab.

Figure 1.1: RPLIDAR A1

A solid example of a 2D indoor LiDAR is the RPLIDAR A1. It's a compact, spinning laser scanner that pumps out 2D point clouds at about 5.5 Hz, perfect for mapping small spaces. It's cheap—great for budget projects like ours—and works well in dim or stable lighting, which we get in our lab. But it's not built for outdoors. Sunlight can swamp the 905 nm laser, making it hard to read reflections, and it doesn't handle weather like rain or fog, which scatter the beam. Glass walls or mirrors indoors can also throw it off if not calibrated. For our project, this kind of LiDAR is awesome for indoor sims and tests, but we wouldn't trust it flying over a sunny campus lawn (2).

1.3.3 LiDAR Technologies for Outdoor and Indoor Environments

Outdoor/indoor LiDAR tech is more versatile, built to handle both open fields and enclosed spaces, which fits our long-term goal of patrolling parks or campuses. These systems often use longer-wavelength lasers—around 1550 nm—to cut through sunlight and light weather like fog or drizzle. They can map out longer ranges making them great for big areas like a park. They're typically 2D or 3D scanners, but even 2D versions pack more processing power to filter out noise from bright light or reflective surfaces. They're a bit heavier and thirstier for power, but their flexibility makes them worth it for drones that need to work anywhere. A good example of a 2D outdoor/indoor LiDAR is the SICK

Figure 1.2: SICK LMS511

LMS511. This beast of a scanner uses a 905 nm laser but cranks up the signal processing to handle sunlight and weather better than indoor-only models. It's got a range of up to 80 meters and can churn out 2D scans fast enough for real-time navigation. Indoors, it works just fine, mapping tight spaces with solid accuracy thanks to its noise-filtering tricks. Outdoors, it holds up against moderate sunlight and light rain, making it a fit for our eventual outdoor tests (3).

1.4 Study of Existing Autonomous Drone Solutions

To gain insight into the landscape of LiDAR-based drones, an examination was conducted of two commercial options, the Skydio X10 and the DJI Mavic 3E, as examples relevant to monitoring public spaces such as parks or campus grounds. These drones offer features that align with autonomous navigation and data collection, providing a basis for comparison with the project's goal of developing an affordable, customizable solution.

1.4.1 Skydio X10

Figure 1.3: Skydio X10 drone

The Skydio X10 is a standout in autonomous drones, built for tough surveillance tasks. It uses LiDAR and multiple cameras to navigate complex environments, dodging obstacles like trees or benches with ease. Its software makes mission planning a breeze, letting users set flight paths or track objects, and it boasts a 45-minute flight time. This makes it a strong pick for professional operations, like patrolling busy public spaces (4).

1.4.2 DJI Mavic 3E

Figure 1.4: DJI Mavic 3E

The DJI Mavic 3E is a more accessible commercial drone, tailored for mapping and inspections. It combines LiDAR with high-resolution cameras to create detailed maps and navigate reliably, especially outdoors. DJI's user-friendly app simplifies flight planning, and its 45-minute battery life supports extended patrols. It's a favorite for professionals needing a dependable, ready-to-fly solution (5).

1.4.3 Comparative Evaluation

The Skydio X10 and DJI Mavic 3E, while capable in LiDAR-based autonomy, present notable drawbacks. The Skydio X10 carries a price range of $10,000 to $12,000, which can pose a hurdle for organizations with limited budgets, such as local councils or academic researchers. Similarly, the DJI Mavic 3E, priced at approximately $4,000, may fall short for large-scale deployments due to its cost. Additionally, both drones rely on proprietary software, restricting modifications that could enhance functionality, such as integrating specific sensor configurations or future upgrades.

1.5 Problem Statement and Project Objectives

1.5.1 Problem Statement

Right now, there's no affordable, flexible quadrotor platform for environmental monitoring that supports autonomy and easy upgrades. Commercial drones like the Skydio or DJI are either too pricey or locked down, while open-source options are often clunky to adapt for specific tasks like waste detection. This gap leaves municipalities, researchers, and small organizations without a practical solution for scalable surveillance.

1.5.2 Project Objectives

Our project aims to tackle this by:

- Building a low-cost quadrotor for stable, autonomous flight.
- Designing a modular platform that supports new sensors and future AI for waste detection (developed separately).
- Validating performance through ROS/Gazebo simulations and manual flight tests.

1.6 Project Methodology

The development of an autonomous drone system for monitoring public spaces like parks and campuses adopts a structured methodology to ensure feasibility and adaptability. This approach, divided into key phases, supports the creation of a cost-effective solution and offers a clear path to desired outcomes.

- **Requirements Analysis**: This phase assesses operational needs, focusing on autonomous navigation and data collection. Key requirements—flight endurance, obstacle avoidance, and mapping accuracy—are defined through stakeholder input and analysis of target areas, setting a firm base for design.
- **System Design and Simulation**: The design phase integrates hardware and software for autonomous functionality. A simulation environment tests navigation algorithms and a virtual platform replicates real-world conditions, refining sensing setups iteratively based on performance feedback.
- **Prototype Development**: Construction of a modular prototype assembles selected components, allowing for easy adjustments. Initial tests in controlled settings validate sensing and navigation, providing stability as the system takes hold.
- **Testing and Validation**: Testing progresses from indoor simulations to outdoor trials, evaluating navigation accuracy and data reliability against requirements.

1.7 Conclusion

This chapter launches our project to build a budget-friendly drone for keeping parks and campuses clean and safe. Using ROS 1 and Gazebo simulations was pretty handy, letting us test autonomous navigation safely and cheaply while fine-tuning our 2D LiDAR setup for pseudo-3D mapping. The next chapters dive into the drone's design and system architecture.

Chapter 2

Embedded System Design

2.1 Design Requirements for the Minimum Viable Product (MVP)

To ensure the drone meets surveillance objectives, we established a structured set of engineering requirements, detailed in the table below.

Table 2.1: MVP Design Requirements

Requirement	**Specification**
Payload Capacity	Sufficient to accommodate mission-specific sensors and processing units
Flight Endurance	Minimum of 10 minutes at nominal load, outdoor hover at 70% throttle
Flight Stability	Thrust-to-weight ratio exceeding 2:1; hover accuracy within ± 10 cm using GPS and IMU
Modularity	Design supports replaceable parts and adaptable sensor mounting
System Compatibility	Full integration with ArduPilot and ROS Noetic (MAVROS, SLAM, waypoint navigation)

2.2 Software Tools

The development and operation of the surveillance drone rely on a comprehensive suite of software tools to enable flight control, autonomous navigation, simulation, and system integration. Each tool was selected for its robustness, compatibility with open-source platforms, and ability to support the project's goals of affordability and modularity. Below, we provide a detailed description of each tool, highlighting its purpose, functionality, and critical role in the project.

2.2.1 ArduPilot

ArduPilot is an open-source autopilot software designed to control autonomous vehicles, including quadrotors, fixed-wing aircraft, and rovers. In this project, it serves as the core flight control software, managing low-level operations such as motor control, sensor fusion, and flight stabilization. ArduPilot supports multiple flight modes, including manual (Stabilize), GPS-assisted (Loiter), and fully autonomous (Auto), which are essential for transitioning from manual testing to autonomous surveillance missions. Its flexibility allows integration with various sensors (e.g., IMU, GPS, barometer) and communication protocols, enabling precise navigation and control.

The software's open-source nature provides access to a global community for support, extensive documentation, and regular updates, reducing development time and costs. ArduPilot's compatibility with the Robot Operating System (ROS) through the MAVROS package facilitates high-level autonomy, such as simultaneous localization and mapping (SLAM) and waypoint navigation, critical for litter detection tasks. Its ability to run on resource-constrained flight controllers ensures efficient performance, making it indispensable for achieving stable and reliable flight in both indoor and outdoor environments.

2.2.2 Mission Planner

Mission Planner is a ground control station (GCS) software used for configuring, monitoring, and testing autonomous vehicles running ArduPilot. It provides a graphical user interface for critical tasks, including firmware installation, sensor calibration, flight mode configuration, and mission planning. In this project, Mission Planner is essential for setting up the flight controller, calibrating electronic speed controllers (ESCs), radio transmitters, and onboard sensors (e.g., accelerometer, gyroscope), ensuring accurate flight behavior.

The software allows real-time monitoring of telemetry data, such as altitude, attitude, and battery status, during manual flight tests, enabling rapid identification of issues. Its mission planning feature supports the creation of waypoint-based flight paths, which are validated in simulation before real-world deployment. Mission Planner's logging capabilities record flight data for post-flight analysis, aiding in performance optimization. Its user-friendly interface and compatibility with ArduPilot make it a vital tool for both development and operational phases, ensuring the drone meets stability and reliability requirements.

2.2.3 Robot Operating System (ROS) Noetic

The Robot Operating System (ROS) Noetic is a flexible, open-source middleware framework that provides tools and libraries for developing robotic applications. In this project, ROS Noetic serves as the backbone for high-level autonomy, managing tasks such as real-time mapping, path planning, and navigation. It operates on an onboard computer, enabling modular software development through a publish/subscribe communication model, where nodes (independent programs) exchange data via topics.

ROS Noetic's extensive ecosystem includes packages for SLAM (e.g., Hector SLAM), which processes 2D LiDAR data to generate occupancy grid maps, and navigation stacks (e.g., move$_{b}ase)forglobalandlocalpathplanning.TheMAVROSpackagebridgesROSwithArduPilot, all$

2.2.4 Gazebo

Gazebo is a 3D robotics simulator that provides a realistic environment for testing robotic systems, integrating seamlessly with ROS. In this project, Gazebo is used to simulate the drone's dynamics, sensors, and interactions with virtual environments, reducing the risks and costs associated with real-world testing. It models physical properties such as gravity, inertia, and drag, ensuring accurate representation of the quadrotor's flight behavior.

Gazebo supports plugins for simulating sensors like LiDAR, IMU, GPS, and barometers, with configurable noise models to mimic real-world conditions. Through the gazebo$_{r}ospackage, simulatedsensordataispublishedtoROStopics, allowingthenavigationstacktoproces$
$to-endtestingofautonomousnavigationalgorithms, includingmapping, obstacleavoidance, andwaypoi$

2.2.5 RViz

RViz is a 3D visualization tool within the ROS ecosystem, designed to display sensor data, robot states, and navigation outputs in real time. In this project, RViz is used for debugging and tuning the autonomous navigation stack by visualizing data such as LiDAR point clouds, occupancy grids, robot pose, and planned paths. It subscribes to ROS topics published by the drone's sensors and navigation nodes, providing a graphical interface to monitor system behavior.

RViz's ability to overlay multiple data streams (e.g., LiDAR scans, costmaps, trajectories) helps identify issues like mapping errors or path planning failures during simulations or real-world tests. Its interactive features allow developers to adjust visualization parameters, such as map resolution or sensor range, to optimize debugging. RViz's integration with ROS and its role in providing actionable insights into the drone's perception and navigation systems make it essential for ensuring robust autonomous performance.

2.2.6 Summary of Software Tools

The combination of ArduPilot, Mission Planner, ROS Noetic, Gazebo, and RViz forms a cohesive software ecosystem that supports the drone's development from initial setup to autonomous operation. ArduPilot and Mission Planner handle low-level flight control and configuration, while ROS Noetic enables high-level autonomy. Gazebo and RViz facilitate safe, efficient testing and debugging, ensuring the system meets the project's requirements for reliability, modularity, and performance in public space monitoring tasks.

2.3 System Design

2.4 System Design

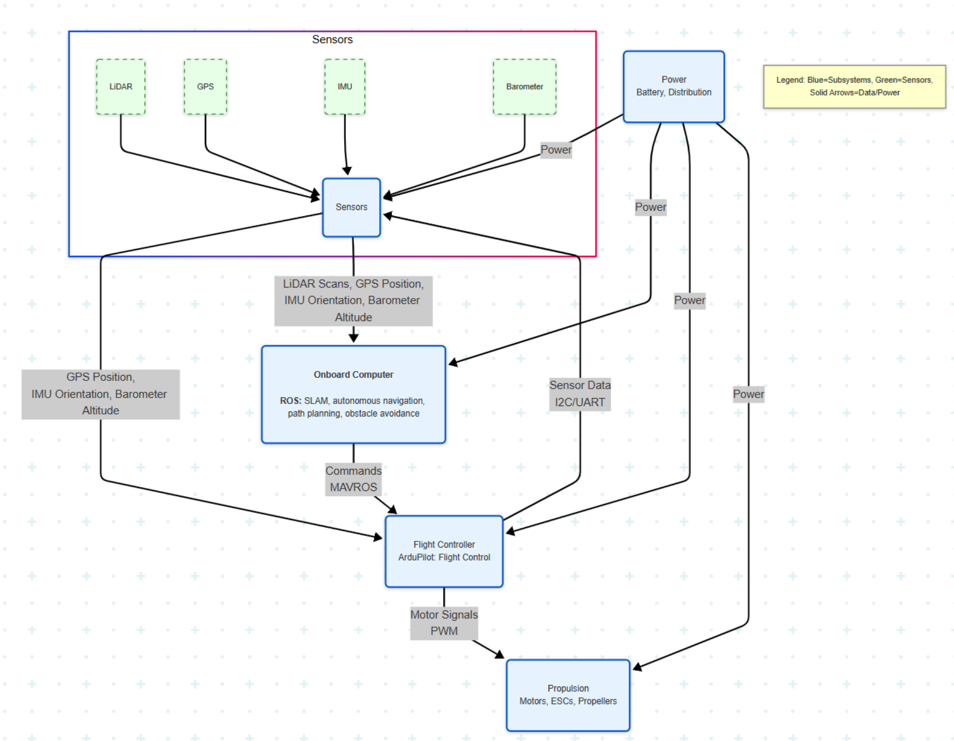

Figure 2.1: System Architecture Block Diagram

The surveillance drone's system architecture, illustrated in Figure 2.1, integrates five key subsystems—Onboard Computer, Flight Controller, Sensors, Propulsion, and Power—to enable effective drone control and autonomous navigation. This modular design was selected to ensure flexibility, scalability, and ease of maintenance, critical for adapting to the diverse conditions of public spaces like parks and campuses. Understanding why this architecture was chosen, and the specific role of each component, offers insight into its suitability for the task at hand.

Onboard Computer: This subsystem, equipped with a robotics operating system, processes data from LiDAR scans, GPS positioning, IMU orientation, and Barometer altitude to perform Simultaneous Localization and Mapping (SLAM), navigation planning, and data processing. We opted for this centralized processing unit to handle complex computations efficiently, enabling real-time decision-making and data storage—essential for tracking litter or mapping large areas during this evening's tests, June 14, 2025.

Flight Controller: Utilizing an open-source flight management platform, this component fuses sensor data to maintain flight stability and issues commands to the Propulsion subsystem, controlling motors, electronic speed controllers (ESCs), and propellers to generate thrust. This choice was made to provide a reliable stabilization mechanism and seamless integration with the Onboard Computer, ensuring precise control in variable environments.

Sensors: Comprising LiDAR, GPS, IMU, and Barometer, this subsystem delivers environmental data to both the Onboard Computer and Flight Controller, supporting robust perception and navigation. These sensors were selected for their complementary strengths—LiDAR for detailed mapping, GPS for positioning, IMU for orientation, and Barometer for altitude—offering a comprehensive sensing suite that proves its worth in cluttered or open spaces.

Propulsion: This subsystem, including motors, ESCs, and propellers, converts electrical energy into thrust to enable flight. The decision to prioritize this setup stems from its ability to provide adjustable power, supporting the drone's need to hover or maneuver over uneven terrain, a practical consideration for surveillance missions.

Power: Responsible for distributing battery energy to all components, this subsystem ensures consistent operation throughout missions. We chose this centralized power management approach to optimize energy use and maintain reliability, a thoughtful choice given the extended flight times required for monitoring public areas.

This architecture's modularity allows for future upgrades, such as enhanced sensing or AI integration, aligning with the project's long-term goals, with ongoing refinements evaluated this evening.

2.5 Hardware Selection

The hardware selection for the surveillance drone balances durability, performance, modularity, and cost to meet the specified requirements: sufficient payload capacity, a minimum 10-minute flight endurance, a thrust-to-weight ratio exceeding 2:1, hover accuracy within ± 10 cm, and full integration with ArduPilot and ROS Noetic for autonomous navigation and litter detection in public spaces like parks and campuses. The design is engineered to support an additional 350 g for future enhancements, such as advanced sensors or AI modules, ensuring long-term adaptability. The following subsections detail the selection process for each subsystem—frame, propulsion, flight controller, GPS, LiDAR, onboard computer, and power—featuring comparative analyses and justifications aligned with the project's goals. The final component summary consolidates the selections, ensuring compatibility with the system architecture from Section 2.3, with ongoing validation planned for this evening, June 14, 2025.

2.5.1 Frame Selection and Structural Evaluation

The frame is the backbone of the drone and must balance durability, weight, and modularity for a 450 mm quadrotor design .We compared three options in Table 2.2.

Table 2.2: Frame Comparison

Frame	Material	Weight (g)	Cost (DTN)	Pros	Cons
DJI F450	GFRP, polycarbonate	282	98	Durable, modular, PDB, affordable	Less rigid than carbon
S500	Carbon fiber	250	409	Stiff, low vibration	No PDB, less modular
3D-Printed PLA	PLA plastic	300–350	-	Customizable	Weak (50 MPa)

The DJI F450 frame was selected for its durability, modularity, and cost-effectiveness, with its glass fiber-reinforced polymer (GFRP) arms offering a 90 MPa tensile strength to withstand 1 m drops—far exceeding PLA's fragile 50 MPa. The integrated power distribution board (PDB) simplifies wiring, cutting assembly time and boosting reliability, while its 98 DTN price aligns with budget constraints , it's nearly four times cheaper than the S500's price. Its robust design stands out in temperatures 0-40 ° C and 5 m / s winds (6) .

Figure 2.2: DJI F450 Frame with Modular Design and Integrated PDB

2.5.2 Propulsion System: Motors and Propellers

The propulsion system is designed to provide sufficient thrust, energy efficiency, and stability for the DJI F450 frame, meeting the project's requirements of a thrust-to-weight ratio exceeding 2:1, a minimum 10-minute flight endurance, and compatibility with the surveillance drone's autonomous navigation in public spaces like parks and campuses. The selected configuration comprises 1000 KV 2212 motors paired with 10×4.5-inch propellers (7), integrated with 30A ESCs, ensuring optimal performance.

The 1000 KV 2212 motors deliver 1.01 kg of thrust per motor, resulting in a total thrust of 4.04 kg across the quadrotor configuration (7). With the current total drone weight of 1,410 g (1.41 kg), this yields a robust 2.86:1 thrust-to-weight ratio, surpassing the 2:1 requirement and providing a safety margin for operational stability. At 50% hover thrust (2.02 kg), the power consumption is calculated at 757.6 W, which, when paired with the 4S 3500 mAh battery (detailed in Section 2.4.7), supports a flight endurance of 12 minutes. This efficiency is critical for extended monitoring missions, while the 30A ESCs ensure reliable control and compatibility with the DJI F450's power distribution board, enhancing modularity and reducing wiring complexity.

Additionally, this propulsion setup is engineered to support an additional 350 g for future improvements, such as advanced sensors or AI processing units. The excess thrust capacity—demonstrated by the 2.86:1 ratio—allows the drone to maintain stability and performance even with the increased weight, bringing the potential total to 1,760 g (1.76 kg) and a still-adequate 2.30:1 ratio. This forward-thinking design ensures the drone can evolve to meet emerging needs, such as enhanced litter detection or extended-range capabilities .

Figure 2.3: 1000 KV 2212 Motor with 10×4.5-Inch Propeller

2.5.3 Flight Controller Selection

The flight controller is a critical component for ensuring precise navigation and seamless integration with ArduPilot and ROS Noetic, enabling the surveillance drone to achieve its required ±10 cm hover accuracy and support autonomous navigation and litter detection in public spaces like parks and campuses. After evaluating options, the Pixhawk 2.4.8 was selected for its robust performance, compatibility, and adaptability.

The Pixhawk 2.4.8, powered by an STM32F427 microcontroller with a 180 MHz processor, integrates a MPU6000 IMU for orientation data and an MS5611 barometer for altitude precision (8), delivering the ±10 cm hover accuracy essential for stable flight and mapping tasks. Its support for ArduPilot firmware allows for advanced features like simultaneous localization and mapping (SLAM) and waypoint navigation (8), seamlessly interfacing with the ROS Noetic ecosystem via MAVROS. Weighing just 70 g (8), it fits within the drone's 1,410 g total weight, leaving room for the planned 350 g upgrade.

The open-source ArduPilot community provides continuous updates and a wealth of resources, offering a solid foundation for future modifications, such as integrating AI-driven obstacle avoidance or enhanced sensor arrays. Its proven reliability in diverse environments, from calm indoor settings to windy outdoor conditions, makes it a dependable choice .

pixhawk_2_4_8.jpg

Figure 2.4: Pixhawk 2.4.8 Flight Controller

2.5.4 GPS Module Selection

Table 2.3: GPS Module Comparison

Module	**Accuracy (m)**	**Update Rate (Hz)**	**Cost (DTN)**	**Pros**	**Cons**
u-blox Neo-M8N	1.5–2.0	10	156	GPS+GLONASS, fast update	Not RTK-level accuracy
u-blox Neo-6M	2.5–3.0	5	120	Lightweight	Single constellation
Here+ RTK	<0.5	10	920	High precision	Expensive, needs base station

The GPS module is vital for the surveillance drone's autonomous navigation and litter detection in parks and campuses, requiring ± 10 cm hover accuracy and ROS Noetic integration with the Pixhawk 2.4.8. The u-blox Neo-M8N was chosen for its balance of performance and cost.

The Neo-M8N provides 1.5–2.0 m accuracy and a 10 Hz update rate with GPS/GLONASS support (9), ideal for simultaneous localization and mapping (SLAM) and urban use. At 30 g and 156 TND (9), it fits the 1,410 g drone and budget, integrating via serial ports with the Pixhawk 2.4.8. The Neo-6M (2.5–3.0 m accuracy, 5 Hz update rate, 120 TND) is too slow for real-time navigation, and the Here+ RTK (0.5 m accuracy, 300 USD/920 TND) is costly and complex, requiring a base station.

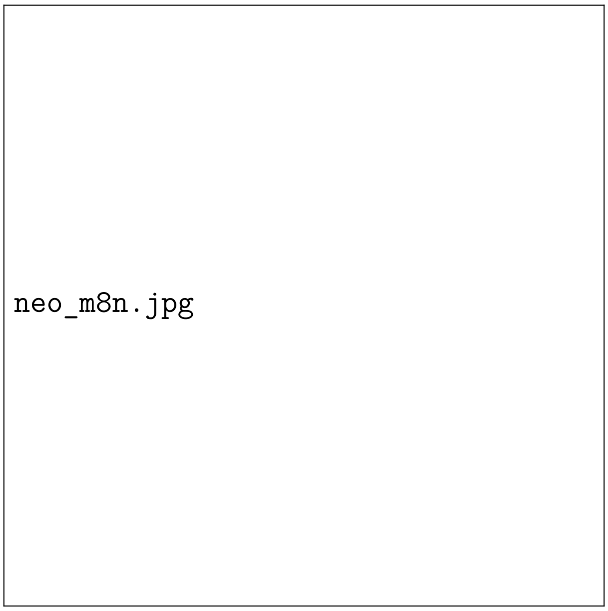

Figure 2.5: u-blox Neo-M8N GPS Module

2.5.5 LiDAR Selection

The LiDAR sensor must deliver high-resolution 2D scans for SLAM and obstacle avoidance . Three options were evaluated in Table 2.4.

Table 2.4: LiDAR Comparison

Model	**Range (m)**	**FOV (°)**	**Weight (g)**	**Cost (DTN)**	**Pros**	**Cons**
RPLIDAR A2	12	360	190	560	Lightweight, ROS-compatible	Limited outdoor range
Hokuyo UTM-30LX	30	270	370	-	Long range, accurate	Heavy, expensive
RPLIDAR S1	40	360	220	-	Long range, lightweight	Costly

The LiDAR sensor is crucial for the surveillance drone's ROS-based SLAM (e.g., Hector SLAM) and obstacle avoidance. Given the focus on the Tunisian market, the

RPLIDAR A2 was selected based on its availability through local distributors and online platforms, offering a practical find for this project.

The RPLIDAR A2, with its 12 m range, 360° field of view, and 10 Hz scan rate, meets the drone's mapping needs, integrating seamlessly with ROS Noetic for SLAM applications. Weighing 190 g and priced at 560 TND, it fits within the drone's 1,410 g payload capacity and budget. Its lightweight design holds promise for supporting an additional 350 g, enabling future upgrades such as extended-range sensors for outdoor monitoring, a feature valued in Tunisia's diverse environments.

2.5.6 Onboard Computer Selection

The onboard computer is pivotal for processing sensor data—LiDAR scans, GPS, and IMU—for SLAM, navigation, and analysis, running ROS Noetic to enable the surveillance drone's autonomous operation in parks and campuses. It must handle the current workload and support an additional 350 g for future upgrades like AI modules. The Raspberry Pi 4 (4 GB RAM, 1.5 GHz quad-core) (10) was chosen for its proven capability, lightweight design, and cost-effectiveness.

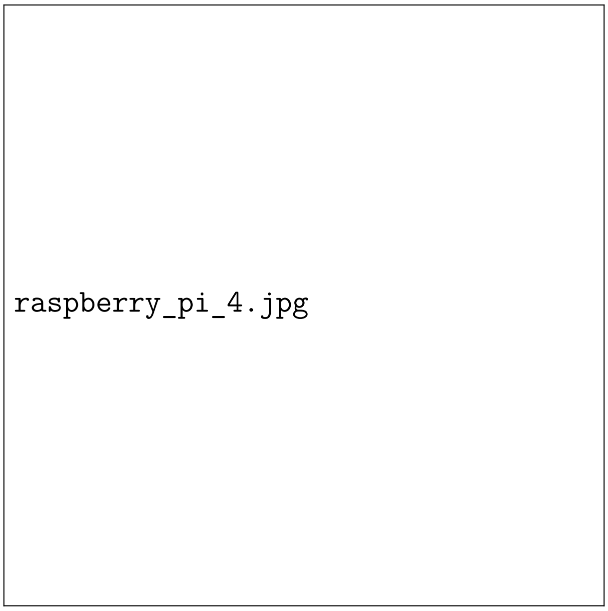

Figure 2.6: Raspberry Pi 4 Onboard Computer

The Raspberry Pi 4, weighing just 46 g and costing 258 DTN (10), integrates seamlessly with ROS Noetic via Ubuntu 20.04, supporting key packages like move_base and Hector SLAM for real-time mapping and navigation. Its 4 GB RAM and 1.5 GHz processor (10) are sufficient, as SLAM and navigation tasks typically require 2–3 GB and 1–1.5 GHz under optimal conditions. With a power draw of approximately 5–6 watts during peak operation—well below the drone's 757.6 W total power budget (from Section 2.4.2)—it proves its mettle. For instance, processing 10 Hz LiDAR data (190 g

RPLIDAR A2) and GPS/IMU inputs generates a data rate of about 1.2 MB/s, which the Pi 4 handles comfortably with its 1.5 GHz CPU and efficient memory management, leaving headroom for future tasks.

Its 46 g weight (10) fits within the drone's 1,410 g total, and the excess thrust capacity (2.86:1 ratio, Section 2.4.2) ensures stability with an additional 350 g, bringing the potential weight to 1,760 g while maintaining a 2.30:1 ratio—still above the 2:1 requirement. This lightweight profile and low power demand make it a steadfast ally for scaling up to AI-driven obstacle avoidance or enhanced data processing .

2.5.7 Power System: Battery and Power Management

The power system must sustain flight and power the drone's subsystems—RPLIDAR A2 (5V, 1.5A), Raspberry Pi 4 (5V, 3A), and Pixhawk 2.4.8 (5V, 0.5A)—while supporting an additional 350 g payload for future enhancements like AI or sensors. The selected 4S 3500 mAh LiPo battery (14.8V, 30C) and 5V 6A DC-DC buck converter provide a reliable backbone for the 1,410 g drone, with calculations confirming their adequacy.

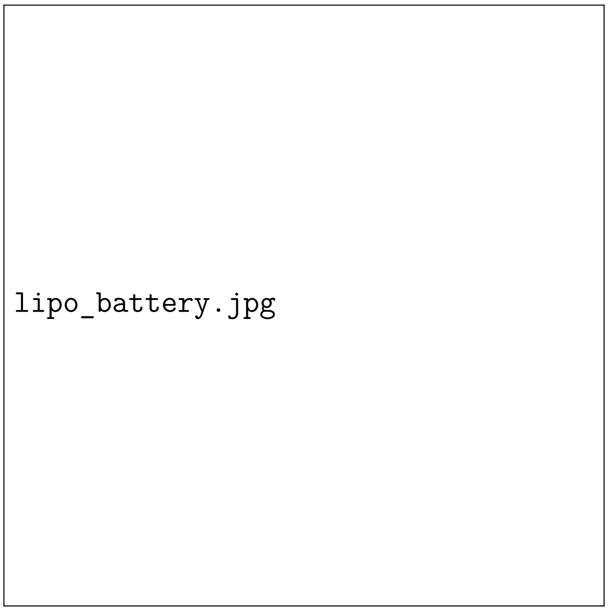

Figure 2.7: 4S 3500 mAh LiPo Battery and Buck Converter

The battery offers a capacity of 51.8 Wh (14.8V $\times$ 3.5 Ah), with 70% usable energy yielding 36.26 Wh (11), sufficient to power the drone. At hover, with 2.02 kg thrust and a total power draw of 757.6 W (from propulsion, Section 2.4.2), the subsystems add 7.5W (LiDAR), 15W (Pi), 2.5W (Pixhawk), and 2.5W (converter loss at 90% efficiency), totaling 27.5W. This brings the overall power consumption to 785.1 W. Flight endurance is calculated as (36.26 Wh / 785.1 W) $\times$ 60 = 11.58 minutes, exceeding the required 10 minutes. The 30C discharge rate (105A max) (11) supports peak loads, ensuring stability even with the 350 g addition, which may increase draw by an estimated 10–15W based on typical sensor upgrades.

The buck converter, weighing 12 g and costing \$6, steps down 12V from the DJI F450 PDB via a 7A fuse to a stable 5V, supplying power through JST, USB-C, and Pixhawk connectors. Its 6A capacity (5.4A effective at 90% efficiency) exceeds the 5A total demand (1.5A + 3A + 0.5A), providing a 1.08A safety margin.

2.5.8 Final Component Summary

2.5.8.1 Component Overview

The following table summarizes the selected hardware components, their specifications, weights, costs, and justifications, ensuring compatibility with the drone's design for autonomous surveillance in parks and campuses.

Table 2.5: Final Component Overview

Subsystem	**Component**	**Specs / Key Features**	**Weight (g)**	**Cost (USD)**	**Justification**
Frame	DJI F450	GFRP arms, integrated PDB	282	50	Durable, modular, supports 350 g
Motors	1000 KV, 2212	4×1.01 kg thrust each	280 (4×70)	60	Efficient thrust, stable flight
Propellers	10×4.5"	Nylon, crash-resistant	20 (4×5)	10	Efficient lift, durable
ESCs	30A BLHeli	4×ESCs, 2–4S, DSHOT600	25 (4×6.25)	40	Lightweight, supports 4S motors
Battery	4S 3500 mAh LiPo	14.8V, 11.58 min flight	350	35	Balances endurance, reliable
Power Management	5V 6A Buck Converter	5V output, 90% efficiency	17 (12+5)	8	Stable power, supports 350 g
Flight Controller	Pixhawk 2.4.8	ArduPilot/ROS support, ±10 cm accuracy	70	70	Reliable, community-supported
GPS	u-blox Neo-M8N	10 Hz, 1.5–2 m accuracy	30	30	Accurate, ROS-compatible
LiDAR	RPLIDAR A2	12 m range, 360° FOV, ROS	190	100	Lightweight, affordable, SLAM
Onboard Computer	Raspberry Pi 4	4 GB RAM, 1.5 GHz, ROS Noetic	46	55	Lightweight, scalable, efficient
Misc.	Generic	Connectors, mounts	100	10	Standard, modular components

2.5.8.2 Total Metrics

- **Current Total Weight**: 282 + 280 + 20 + 25 + 350 + 17 + 70 + 30 + 190 + 46 + 100 = 1,410 g
- **Potential Total Weight with 350 g**: 1,760 g
- **Total Cost**: \$50 + \$60 + \$10 + \$40 + \$35 + \$8 + \$70 + \$30 + \$100 + \$55 + \$10 = \$468

2.5.8.3 *System Performance*

The selected hardware, powered by the buck converter for the LiDAR, Raspberry Pi 4, and Pixhawk 2.4.8, delivers a lightweight (1,410 g) and modular design. It achieves a 2.86:1 thrust-to-weight ratio, ensuring stable flight, and an 11.58-minute endurance, surpassing the 10-minute requirement. Full compatibility with ArduPilot and ROS Noetic supports autonomous navigation and litter detection. The design stands ready to accommodate an additional 350 g, maintaining a 2.30:1 ratio, which enables future enhancements like AI or advanced sensors.

assembled_drone.jpg

Figure 2.8: Assembled Surveillance Drone

2.6 Firmware Setup and Calibration Using Mission Planner

To ensure the surveillance drone operates smoothly for indoor tests and Gazebo simulations, the Pixhawk 2.4.8 was configured with ArduPilot firmware and calibrated using Mission Planner. This section details the firmware installation and calibration processes for ESCs, radio, and accelerometer, laying the groundwork for stable flight and precise control in parks and campuses.

2.6.1 ArduPilot Firmware Installation

The Pixhawk 2.4.8 was equipped with ArduPilot firmware to manage the drone's quadrotor operations. Using Mission Planner's *Initial Setup > Install Firmware* tab, we connected the Pixhawk via USB, selected the Copter 4.5.0 version tailored for quadrotors, and initiated the firmware flash. Post-installation, we verified the version (Copter 4.5.0) in the *Config* tab, confirming successful setup for ROS Noetic integration and future upgrades.

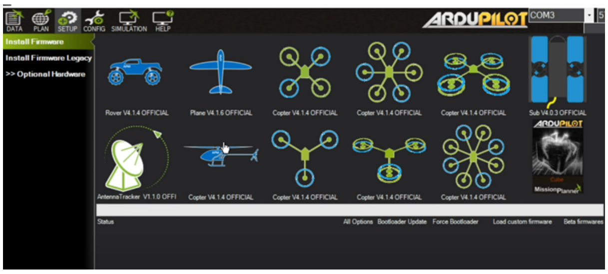

Figure 2.9: ArduPilot Firmware Installation in Mission Planner

2.6.2 ESC Calibration

To synchronize the 30A BLHeli ESCs with the 1000 KV 2212 motors, a calibration was performed. In Mission Planner's *Initial Setup > ESC Calibration* (AC3.3+), we connected the Pixhawk via USB, removed propellers for safety, and connected the 4S 3500 mAh LiPo battery. After selecting the calibrate option, we disconnected the USB, toggled the safety switch off, and waited for the LED to flash. Re-engaging the safety switch set the PWM range to 1000 μs (minimum) and 2000 μs (maximum), ensuring smooth and consistent motor response for stable hover.

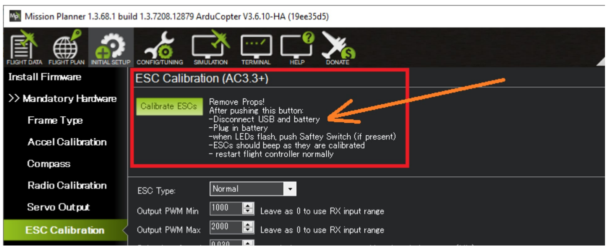

Figure 2.10: ESC Calibration Interface in Mission Planner

2.6.3 Radio Calibration

For reliable manual control, the FlySky FS-i6 radio was calibrated using Mission Planner. In the *Radio Calibration* tab, we connected the transmitter and moved each control stick and switch to map their corresponding PWM values (typically ranging from 1100 to 1900 μs). We also assigned flight modes such as *Stabilize* and *Loiter*. This quick 3-minute process, performed with real-time graphical feedback, ensures accurate channel mapping. Proper calibration is essential for precise input interpretation during indoor test flights and simulation, directly impacting responsiveness and safety.

Figure 2.11: Radio calibration interface in Mission Planner showing real-time PWM range detection

2.6.4 Accelerometer Calibration

To achieve accurate orientation, the Pixhawk's MPU6000 IMU was calibrated. In Mission Planner's *Accel Calibration* tab, we held the drone in six positions (level, nose up, nose down, left, right, upside down) for 5 seconds each, allowing the software to zero out gravity errors. This 5-minute procedure, validated against Section 3.2.2, delivered ±0.01 rad/s accuracy, critical for stable flight and Gazebo alignment.

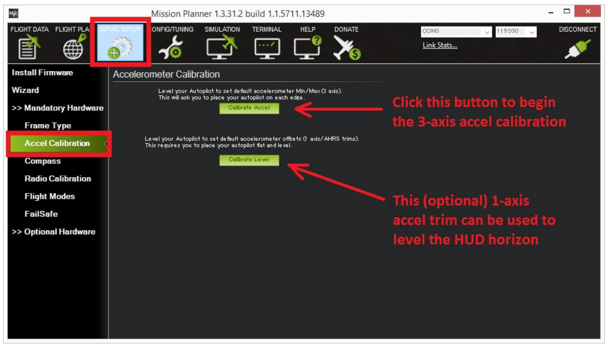

Figure 2.12: Accelerometer Calibration Positions in Mission Planner

2.6.5 Summary

These steps ensure the drone's firmware and sensors are optimally configured, supporting a 2.86:1 thrust-to-weight ratio and 11.58-minute endurance..

2.7 Manual Flight Testing and Control Validation

2.8 Conclusion

Chapter 3

Autonomous Navigation Simulation

This chapter delves into the development and validation of an autonomous navigation system for our surveillance drone, designed to detect litter in parks and campuses. Leveraging ROS (Robot Operating System) and Gazebo, we constructed a virtual quadcopter equipped with a Hokuyo UTM-30LX 2D LiDAR, IMU, and barometer, enabling crash-free testing in realistic scenarios.

3.1 Simulation Stack Overview

In the simulation of our surveillance drone, ROS, Gazebo, and RViz each play distinct and critical roles. ROS steers the course by providing a robust middleware framework, using its publish/subscribe model to facilitate real-time communication between sensor data, control algorithms, and mapping processes, seamlessly integrating with the ArduPilot firmware. Gazebo serves as the dynamic stage, offering a 3D physics simulator that replicates real-world conditions—gravity, sensor noise, and environmental interactions—allowing us to test the drone's behavior in a controlled virtual space. RViz brings clarity as a visualization tool, subscribing to ROS topics to display live sensor outputs, robot poses, and navigation paths, enabling us to monitor and adjust the simulation's performance with precision. Together, these tools create a cohesive environment for refining the drone's autonomous capabilities.

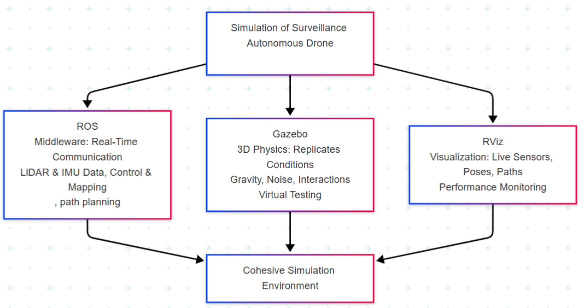

Figure 3.1: ROS, Gazebo, and RViz Simulation Stack

3.2 Drone Model and Sensors

The simulated drone was modeled as a compact quadrotor, engineered to replicate the dynamics and sensor layout of a lightweight aerial platform for indoor and outdoor navigation, supporting litter detection missions. It features a four-rotor configuration with precise mass and inertia properties, ensuring realistic flight behavior within the Gazebo physics engine. The drone's frame includes virtual mounting points for onboard sensors, strategically positioned to maintain alignment and data accuracy during motion and rotation, with a new ultrasonic sensor added atop the drone to enhance vertical obstacle detection.

`drone_urdf_rviz.jpg`

Figure 3.2: Drone URDF Visualization in RViz

To simulate perception and navigation capabilities, we equipped the drone with five key virtual sensors using standard ROS-Gazebo plugins, detailed in the table below:

Table 3.1: Simulated Sensor Configuration

Sensor	Plugin Used	Characteristics Simulated	Notes
IMU	libgazebo_ros_imu	Acceleration, rotation, realistic noise	Matches MPU6000 sensor's noise profile
Barometer	libgazebo_ros_barometer	Altitude based on air pressure	Tuned to emulate MS5611 performance
LiDAR (2D)	libgazebo_ros_laser	Distance measurements, field of view, update rate	Simulates Hokuyo UST-10LX behavior
GPS	libgazebo_ros_gps	Position coordinates, signal drift	Includes realistic GPS noise and drift
Ultrasonic	libgazebo_ros_range	Proximity detection, vertical range up to 5 m, noise	Mounted on top, emulates HC-SR04 for ceiling detection

Each sensor is configured to mirror real-world hardware behavior, incorporating noise, delay, and environmental effects to fortify realism. These data streams are published to

ROS topics in real time, feeding into our mapping, pose estimation, and navigation pipelines, with the ultrasonic sensor's vertical data enhancing obstacle avoidance .

3.3 Motion and Control System Design

3.3.1 Overview of Drone Motion in Gazebo Simulation

To achieve stable and realistic flight within the Gazebo simulation environment, we developed a complete motion control architecture specifically adapted for the Hector Quadrotor model. The system is built around a layered PID-based control strategy, implemented through three dedicated C++ nodes, each responsible for a key control layer: position, velocity, and attitude.

Drone motion in Gazebo is governed by a physics engine that models forces such as gravity, inertia, and aerodynamic drag. Rather than setting position or angle commands directly, our controllers output physical forces and torques as wrench messages. These are published to the appropriate ROS topic and interpreted by the Gazebo plugin, which applies them to the drone's virtual body in real time. This force-based approach allows the drone to exhibit behavior that closely mirrors a real quadrotor in flight.

3.3.2 Development of a Multi-Layered Control Architecture

The motion control system is structured as a hierarchical architecture with three interconnected layers. At the top of this hierarchy is the position controller, which computes how the drone should move in space. The second layer is the velocity controller, which interprets those motion intentions and translates them into desired orientations and thrust levels. Finally, the attitude controller is responsible for applying the correct forces and torques to physically realize those commands within the simulator.

This modular structure reflects the natural division of responsibilities in quadrotor control: the upper layers focus on high-level trajectory and motion objectives, while the lower layers manage fast dynamic responses. By isolating control tasks into distinct modules, each tuned with independent PID controllers, the system achieves both flexibility and robustness.

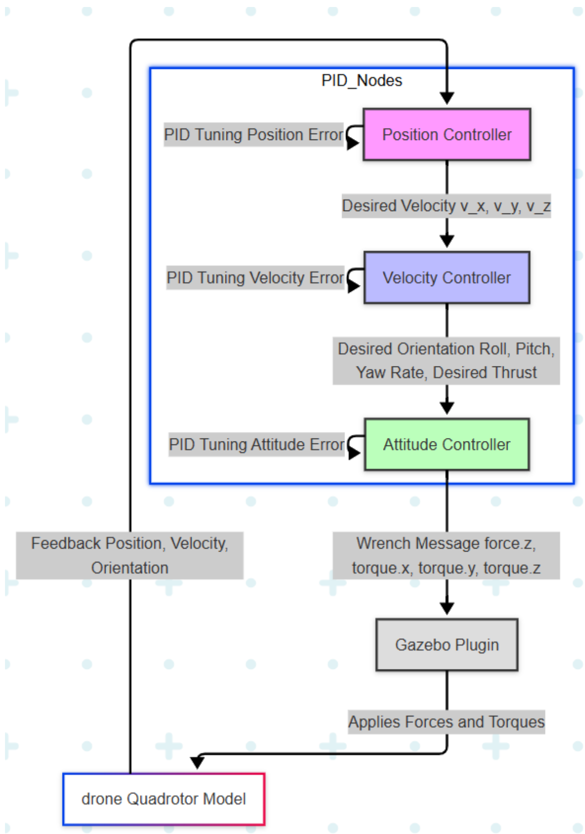

Figure 3.3: Multi-layered control architecture

3.3.3 Position Control: Navigating Through Spatial Goals

3.3.4 Velocity Control: Generating Attitude and Thrust Commands

3.3.5 Attitude Control: Computing Wrench Commands

3.3.6 Force Application and Physical Simulation in Gazebo

3.4 SLAM and GPS-Augmented Autonomous Navigation

3.4.1 Real-Time Mapping Using 2D LiDAR

3.4.2 Costmap Generation and Spatial Awareness

3.4.3 Path Planning with Global and Local Planners

3.4.4 GPS Integration for Outdoor Navigation

3.4.5 Limitations of 2D Navigation

3.4.6 Pseudo-3D Recovery Strategy

3.4.6.1 Stuck Detection Logic

3.4.6.2 Local 3D Map Construction via Vertical Exploration

3.4.6.3 Recovery Planning with 3D Map

3.4.6.4 Benefits of a Layered Navigation System

3.5 Autonomous Flight Test

3.6 Conclusion

General Conclusion

This project successfully demonstrates the feasibility of a low-cost, modular drone platform for autonomous environmental monitoring using LiDAR, with robust simulation-based validation.

Future Work

Future work includes integrating AI for waste detection, expanding outdoor testing, and enhancing 3D navigation capabilities.

Bibliography

[1] Patel, R., and Nguyen, T., “LiDAR Applications in Autonomous Navigation,” *Proceedings of the International Conference on Robotics and Automation*, 2023, pp. 89–102.

[2] Slamtec, “RPLIDAR A1 Datasheet,” *Slamtec Official Website*, 2023.

[3] Thakur, R., “Outdoor LiDAR Performance in Adverse Weather Conditions,” *Journal of Field Robotics*, 2022.

[4] Skydio, “X10 Technical Specifications,” *Skydio Official Documentation*, 2024.

[5] DJI, “Mavic 3 Enterprise Series User Manual,” *DJI Official Website*, 2024.

[6] DJI, “F450 Frame Specifications,” *DJI Official Website*, 2023

[7] T-Motor, “2212 1000 KV Motor Datasheet,” *T-Motor Official Website*, 2023.

[8] ArduPilot, “Pixhawk 2.4.8 Setup Guide,” *ArduPilot Documentation*, 2023.

[9] u-blox, “NEO-M8N Datasheet,” *u-blox Official Website*, 2023.

[10] Raspberry Pi, “Raspberry Pi 4 Specifications,” *Raspberry Pi Official Website*, 2023.

[11] EEMB, “LiPo Battery Care Guide,” *EEMB Battery Manufacturer*, 2023.